LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-TR-677976

A Bayesian inversion framework for yield and height-of-burst/depth-of-burial for near-surface explosions

G. Johannesson, V. Bulaevskaya, A. Ramirez, S. Ford, A. Rodgers

October 7, 2015

Disclaimer

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

A Bayesian Inversion Framework for Yield and Height-of-Burst/Depth-of-Burial for Near-Surface Explosions

Gardar Johannesson, Vera Bulaevskaya, Abe Ramirez, Sean Ford, Artie Rodgers

Lawrence Livermore National Laboratory

September 29, 2015

Abstract

A Bayesian inversion framework is presented to estimate the yield of an explosion and height-of-burst/depth-of-burial (HOB/DOB) using seismic and air pressure data. This is accomplished by first calibrating the parameters in the forward models that relate the observations to the yield and HOB/DOB and then using the calibrated model to estimate yield and HOB/DOB associated with a new set of seismic and air pressure observations. The MCMC algorithms required to perform these steps are outlined, and the results with real data are shown. Finally, an extension is proposed for a case when clustering in the seismic displacement occurs as a function of different types of rock and other factors.

1 Introduction

The problem presented here is described in detail in Ford et al. (2014). In summary, the goal is to estimate the yield of an explosion and the height-of-burst/depth-of-burial (HOB/DOB) using observed seismic data (in this case the vertical displacement of the first P arrival) and air pressure data (the integral of the overpressure/impulse). Hence, a predictive (forward) model is proposed that depends on the unknown yield and HOB, along with the distance between the explosions and a network of seismic and meteorology stations, plus unknown regression parameters. The forward model is calibrated using explosions with known yield and HOB. Finally, the calibrated model is used to estimate the yield and HOB for an explosion not included in the training data.

2 Notation and General Setup

For any given explosion/event we have two sources of observations: the seismic displacement of the first P arrival at a given network of seismic stations and the integral of the air pressure (impulse) at a given network of meteorology sites.

2.1 Notation

2.1.1 The Events

For the explosions (the events), let

w_i = the yield of the i-th event (in kg),
n_e = the number of events,
h_i = the height of burst (HOB) of the i-th event (in m, with positive and negative values indicating above and below ground, respectively),
$h_i^s = h_i w_i^{-1/3}$ = the scaled HOB.

2.1.2 The Seismic Data

The seismic data consist of the following:

$$
\begin{aligned}
d_{ij} &= y_{dij} = \text{the displacement of the first P arrival for event } i \\
&\quad \text{observed at seismic station } j, \\
\sigma_{dij} &= \text{an estimate of the standard deviation of the observation} \\
&\quad \text{error associated with } y_{dij}, \\
n_d &= \text{the number of seismic stations (displacement observation} \\
&\quad \text{stations}), \\
n_{di} &= \text{the number of displacement observations for event } i, \\
r_{dij} &= \text{the distance between event } i \text{ and seismic station } j, \\
d^s_{ij} &= y^s_{dij} = d_{ij} w_i^{-1/3} = \text{the scaled displacement}, \\
r^s_{dij} &= r_{dij} w_i^{-1/3} = \text{the scaled distance}.
\end{aligned}
$$

Now, if event i is not observed at seismic station j, then d_{ij} is simply missing and further, n_{di} can be equal to zero, meaning no displacement observations are available for event i.

2.1.3 The Air Pressure Data

For the air pressure data, let

$$
\begin{aligned}
a_{ij} &= y_{aij} = \text{the air pressure impulse for event } i \text{ observed at} \\
&\quad \text{meteorology station } j \text{ (in Pa-s)}, \\
\sigma_{aij} &= \text{an estimate of the standard deviation of the observation} \\
&\quad \text{error associated with } y_{aij}, \\
n_a &= \text{the number of meteorology stations}, \\
n_{ai} &= \text{the number of air pressure impulse observations for event} \\
&\quad i, \\
r_{aij} &= \text{the distance between event } i \text{ and meteorology station } j \\
&\quad \text{(in m)}, \\
P_{ij} &= \text{the air pressure associated with observation } a_{ij} \text{ (in Pa)}, \\
T_{ij} &= \text{the temperature associated with observation } a_{ij} \text{ (in K)}, \\
a^s_{ij} &= y^s_{aij} = a_{ij} w^{-1/3} (P_{ij}/P_0)^{-2/3} (T_{ij}/T_0)^{1/2} = \text{the scaled im-} \\
&\quad \text{pulse, where } P_0 = 101,325 \text{ Pa and } T_0 = 288\text{K}, \\
r^s_{aij} &= r_{aij} w^{-1/3} (P_{ij}/P_0)^{1/3} = \text{the scaled distance}.
\end{aligned}
$$

Note that the index j is different from the index j for the seismic data (it can be confusing, but it simplifies notation). Similarly to the seismic data,

some of the a_{ij}'s can be missing and if $n_{ai} = 0$, then we do not have any impulse observations for event i.

2.2 The Forward Models

The broader empirical relationship that is assumed to hold for the displacement observations is

$$\begin{aligned}\log_{10}(d_{ij}/w_i^{1/3}) = \beta_{d1} + \beta_{d2}\log_{10}(r_{dij}/w_i^{1/3}) + \\ \beta_{d3}\tanh(\beta_{d4} + \beta_{d5}(h_i/w_i^{1/3})).\end{aligned} \tag{1}$$

Equivalently,

$$\log_{10}(d_{ij}^s) = \beta_{d1} + \beta_{d2}\log_{10}(r_{dij}^s) + \beta_{d3}\tanh(\beta_{d4} + \beta_{d5}h_i^s). \tag{2}$$

For the impulse observations,

$$\log_{10}(a_{ij}^s) = \beta_{a1} + \beta_{a2}\log_{10}(r_{aij}^s) + \beta_{a3}h_i^s + \log_{10}(1 + 10^{10\beta_{a3}h_i^s})/10. \tag{3}$$

2.3 Bayesian Inference

The goal here is not to provide a comprehensive introduction to the Bayesian approach to statistical inference, but rather give a brief introduction.

Within the context of the above observations and forward models, let

$$p(y \,|\, \hat{y}(\beta), \kappa) = \text{conditional probability density function (PDF)},$$

where y is a vector of observations, $\hat{y}(\beta)$ is a vector of predictions of the observation values for a given input parameter vector β, and κ is a parameter that captures the accuracy of the prediction. For example, one way to model the PDF is as follows:

$$p(y \,|\, \hat{y}(\beta), \kappa) = \prod_{i=1}^{n} \text{Gau}(y_i \,|\, \hat{y}_i(\beta), 1/\kappa), \tag{4}$$

where

$$\text{Gau}(y \,|\, \hat{y}, 1/\kappa) = (2\pi)^{-1/2}\kappa^{1/2}e^{-\frac{1}{2}\kappa(y-\hat{y})^2}$$

denotes the Gaussian/normal distribution with mean $\hat{y}$ and variance $1/\kappa$. The PDF in (4) is also often referred to as the likelihood function, carrying the information about the "likelihood" of various configurations of β and κ given a realization of the observations.

Given the observations, our goal is to make inference on β and κ. Within the Bayesian framework both β and κ are treated as stochastic/random quantities which we assume follow some *prior* distributions, $p(\beta)$ and $p(\kappa)$. For a concrete example (related to our application), let

$$\begin{aligned} p(\beta) &= p(\beta_1, \ldots, \beta_m) = \prod_{j=1}^{m} \mathrm{Gau}(\beta_j \,|\, M_j, V_j), \\ p(\kappa) &= \mathrm{Gam}(\kappa \,|\, A, B), \end{aligned} \tag{5}$$

where m is the dimension of the input vector β, and

$$\mathrm{Gam}(x \,|\, \alpha, \delta) = \frac{\delta^\alpha}{\Gamma(\delta)} x^{\alpha-1} e^{-\delta x}$$

denotes a gamma distribution with a shape parameter α and a rate parameter δ (i.e., a gamma distribution with mean equal to α/δ and variance equal to α/δ^2).

Our goal is then to derive the *posterior* distribution,

$$\begin{aligned} \pi(\beta, \kappa) \equiv p(\beta, \kappa \,|\, y) &= \frac{p(y \,|\, \hat{y}(\beta), \kappa) p(\beta) p(\kappa)}{p(y)} \\ &\propto p(y \,|\, \hat{y}(\beta), \kappa) p(\beta) p(\kappa), \end{aligned} \tag{6}$$

where $p(y) = \int p(y \,|\, \hat{y}(\beta), \kappa) p(d\beta) p(d\kappa)$ is the marginal distribution of the data and is just a constant given a realization/observation of the data (i.e., it does not depend on β or κ).

Typically, the above posterior distribution is not available in closed form, as an analytical expression. A common approach to explore the posterior distribution is to generate realizations from it and use those to compute the statistics of interest (say mean and variance). For example, this can be accomplished with Markov chain Monte Carlo (MCMC) samplers (e.g., see Gelman et al. (2013) and Robert and Casella (2004)), which are briefly reviewed next.

2.4 Basic MCMC Samplers

The MCMC sampler generates realizations from a Markov chain using a *transition kernel*, $K(\cdot, \cdot)$, such that given the current state of the Markov chain, say $x^{(t)}$, (which can be either a scalar or a vector), the next state is generated as $x^{(t+1)} \sim K(x^{(t)}, \cdot)$, where the transition kernel $K(x, \cdot)$ is a conditional PDF and $x \sim p(\cdot)$ means "x is distributed according to the PDF $p(\cdot)$".

We will now describe two methods to construct the needed transition kernels.

Metropolis-Hastings (MH) Posterior Samplers

The Metropolis-Hastings (MH) algorithm is a method of constructing transition kernels that yield MCMC samples from a given target distribution, which in our case is the posterior distribution of interest. The algoritm uses a *proposal distribution*, $q(x^{(t+1)} \,|\, x^{(t)})$, to generate a candidate point, $x^* \sim q(\cdot \,|\, x^{(t)})$, to be accepted or rejected as the next point in the Markov chain. The general algorithm is given in Algorithm 1.

Algorithm 1 The Metropolis-Hastings sampling algorithm.

Let $\pi(x)$ be the target distribution of interest, $q(y \,|\, x)$ the conditional proposal distribution, and $x^{(t)}$ the current state of the Markov chain. The following procedure yields the next state of the Markov chain:

1: **procedure** MetropolisHastings($x^{(t)}$, $\pi(\cdot)$, $q(\cdot \,|\, \cdot)$)
2: Generate $x^* \sim q(\cdot \,|\, x^{(t)})$
3: Let

$$\alpha(x^*, x^{(t)}) = \frac{\pi(x^*)}{\pi(x^{(t)})} \frac{q(x^{(t)} \,|\, x^*)}{q(x^* \,|\, x^{(t)})}$$

Equivalently

$$\log \alpha(x^*, x^{(t)}) = (\log \pi(x^*) - \log \pi(x^{(t)})) - (\log q(x^* \,|\, x^{(t)}) - \log q(x^{(t)} \,|\, x^*))$$

4: Generate $u \sim \text{Unif}(0, 1)$ [a uniform PDF between 0 and 1]
5: **if** $\alpha(x^*, x^{(t)}) \geq u$ **then** $x^{(t+1)} \leftarrow x^*$ **else** $x^{(t+1)} \leftarrow x^{(t)}$
6: **return** $x^{(t+1)}$
7: **end procedure**

A popular choice for a proposal distribution is a random walk, $x^* = x^{(t)} + \epsilon^{(t)}$, where $\epsilon^{(t)} \sim q(\cdot)$. An example is a normal random walk, with $q(\epsilon) = \text{Gau}(\epsilon \,|\, 0, \tau^2)$, which we can also write as $q(x^* \,|\, x^{(t)}) = \text{Gau}(x^* \,|\, x^{(t)}, \tau^2)$. Note that the normal random walk is symmetric, meaning that $q(x^* \,|\, x^{(t)}) = q(x^{(t)} \,|\, x^*)$, in which case $\alpha(x^*, x^{(t)}) = \pi(x^*)/\pi(x^{(t)})$ in Algorithm 1.

To be more concrete, if we assume the likelihood in (4) and the priors in (5), then a random walk MH algorithm to update any of the $\beta_1, \ldots, \beta_m$ is given in Algorithm 2. Similarly, Algorithm 3 provides a pseudo code for updating κ using MH random walk. Note that in sampling $\kappa \geq 0$, a normal random walk is used that "bounces" off of zero; that is, if the new

proposal $\kappa^* < 0$, then $\kappa^* \leftarrow -\kappa^*$. Finally, there is nothing that restricts the MH algorithm to just update a single parameter at a time. If some of the parameters to be sampled are highly correlated, it is then often better to update the highly correlated parameters jointly. Algorithm 4 outlines an MH multivariate random walk algorithm to update the entire β vector jointly (of course, it could also just be limited to a subset of the β vector, i.e., the parameters that are highly correlated).

Algorithm 2 An MH random walk algorithm for β_j.

1: **procedure** MH_RW_$\beta(j,\, \beta^{(t)},\, \kappa^{(t)};\, \tau)$
2: Let $\beta^* \leftarrow \beta^{(t)} = (\beta_1^{(t)}, \ldots, \beta_m^{(t)})$
3: Only change the j-th element using random-walk, $\beta_j^* \leftarrow \beta_j^{(t)} + \epsilon$, where $\epsilon \sim \text{Gau}(0, \tau^2)$, with τ provided (the "step-size")
4: Compute $\hat{y}(\beta^*)$, reflecting the change in β_j^*
5: Compute the log-acceptance rate,

$$\begin{aligned}
\log \alpha(\beta_j^*, \beta_j^{(t)}) &= \sum_{i=1}^{n} \Big(\log \text{Gau}(y_i \,|\, \hat{y}_i(\beta^*), 1/\kappa^{(t)}) - \log \text{Gau}(y_i \,|\, \hat{y}_i(\beta^{(t)}), 1/\kappa^{(t)}) \Big) \\
&\quad + \Big(\log \text{Gau}(\beta_j^* \,|\, M_j, V_j) - \log \text{Gau}(\beta_j^{(t)} \,|\, M_j, V_j) \Big) \\
&= -\frac{\kappa^{(t)}}{2} \sum_{i=1}^{n} \Big((y_i - \hat{y}_i(\beta^*))^2 - (y_i - \hat{y}_i(\beta^{(t)}))^2 \Big) \\
&\quad - \frac{1}{2V_j} \Big((\beta_j^* - M_j)^2 - (\beta_j^{(t)} - M_j)^2 \Big)
\end{aligned}$$

6: Generate $u \sim \text{Unif}(0, 1)$
7: **if** $\alpha(\beta_j^*, \beta_j^{(t)}) \geq u$ **then** $\beta_j^{(t+1)} \leftarrow \beta_j^*$ **else** $\beta_j^{(t+1)} \leftarrow \beta_j^{(t)}$
8: **return** $\beta_j^{(t+1)}$
9: **end procedure**

Algorithm 5 outlines an MH algorithm to sample from $p(\beta, \kappa \,|\, y)$, by stepping through MH random walk updates for the β_j's and then κ within each MCMC iteration. Note that each β_j and κ has its own "step-size" ($\tau_{\beta j}$ and τ_κ, respectively). The sequence of β_j updates could be replaced with a single joint update of the entire β vector.

The performance/efficiency of the MH normal random walk algorithm in Algorithms 2, 3, and 4 is closely tied to the value of variance/covariance

Algorithm 3 An MH random walk algorithm for κ.

procedure MH_RW_$\kappa(\beta^{(t)}, \kappa^{(t)}; \tau)$

Let $\kappa^* \leftarrow \kappa^{(t)} + \epsilon$, where $\epsilon \sim \text{Gau}(0, \tau^2)$. If $\kappa^* < 0$, then $\kappa^* \leftarrow -\kappa^*$

Compute the log-acceptance rate,

$$\begin{aligned}\log \alpha(\kappa^*, \kappa^{(t)}) &= \sum_{i=1}^{n} \Big(\log \text{Gau}(y_i \mid \hat{y}_i(\beta^{(t)}), 1/\kappa^*) - \log \text{Gau}(y_i \mid \hat{y}_i(\beta^{(t)}), 1/\kappa^{(t)}) \Big) \\ &\quad + \Big(\log \text{Gam}(\kappa^* \mid A, B) - \log \text{Gam}(\kappa^{(t)} \mid A, B) \Big) \\ &= (n/2 + A - 1)(\log \kappa^* - \log \kappa^{(t)}) - (nS^2/2 + B)(\kappa^* - \kappa^{(t)})\end{aligned}$$

where $S = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}(\beta^{(t)}))^2}$

Generate $u \sim \text{Unif}(0, 1)$

if $\alpha(\kappa^*, \kappa^{(t)}) \geq u$ **then** $\kappa^{(t+1)} \leftarrow \kappa^*$ **else** $\kappa^{(t+1)} \leftarrow \kappa^{(t)}$

return $\kappa_j^{(t+1)}$

end procedure

driving the random walk (the τ's and the Σ). If it is too small, the new proposed state will be very similar to the current state and will be accepted with a high probability, yielding a chain that "moves" slowly and most likely needs a larger number of iterations to explore the posterior space. On the other hand, if τ or Σ is too big, many proposals will be rejected, yielding an inefficient random walk that again needs many iterations to explore the posterior space. There are various ways to adaptively pick a "good" value of τ and Σ. One is simply by trial and error, guided by aiming for an acceptance rate of, say, 50% for a univariate sampler (this is considered a good acceptance rate for a univariate random walk sampler). Algorithm 6, on the other hand, gives a simple adaptive algorithm that adjusts τ_κ on the fly when sampling κ in the MCMC Algorithm 3 (see Andrieu and Thoms (2008) for a tutorial on adaptive MCMC algorithms). In short, the adaptation algorithm learns what the variance of the posterior distribution of κ is on the fly and takes the variance of the random walk proposal distribution to be proportional to it. The proportionality constant is taken to be $\lambda^2 = 2.4^2$, based on empirical results when the actual posterior distribution is normal, in which case it yields an acceptance rate of roughly 55%. In practice, one does not necessarily start the adaptation right away (at $t = 1$) since the initial value of the chain might be somewhat far away from the main posterior mass (i.e. way out in the tail of the posterior

Algorithm 4 An MH multivariate random walk algorithm for β.

procedure MH_MRW_$\beta(\beta^{(t)}, \kappa^{(t)}; \Sigma)$

Let $\beta^* \leftarrow \beta^{(t)} + \epsilon$, where $\epsilon \sim \text{Gau}_m(0, \Sigma)$ and $\text{Gau}_m(\mu, \Sigma)$ denotes an m-dimensional multivariate Gaussian distribution with mean vector μ and covariance matrix Σ

Compute $\hat{y}(\beta^*)$

Compute the log-acceptance rate,

$$\begin{aligned}
\log \alpha(\beta^*, \beta^{(t)}) &= \sum_{i=1}^{n} \Big(\log \text{Gau}(y_i \,|\, \hat{y}_i(\beta^*), 1/\kappa^{(t)}) - \log \text{Gau}(y_i \,|\, \hat{y}_i(\beta^{(t)}), 1/\kappa^{(t)}) \Big) \\
&\quad + \sum_{j=1}^{m} \Big(\log \text{Gau}(\beta_j^* \,|\, M_j, V_j) - \log \text{Gau}(\beta_j^{(t)} \,|\, M_j, V_j) \Big) \\
&= -\frac{\kappa^{(t)}}{2} \sum_{i=1}^{n} \Big((y_i - \hat{y}_i(\beta^*))^2 - (y_i - \hat{y}_i(\beta^{(t)}))^2 \Big) \\
&\quad - \sum_{j=1}^{m} \frac{1}{2V_j} \Big((\beta_j^* - M_j)^2 - (\beta_j^{(t)} - M_j)^2 \Big)
\end{aligned}$$

Generate $u \sim \text{Unif}(0, 1)$

if $\alpha(\beta^*, \beta^{(t)}) \geq u$ **then** $\beta^{(t+1)} \leftarrow \beta^*$ **else** $\beta^{(t+1)} \leftarrow \beta^{(t)}$

return $\beta^{(t+1)}$

end procedure

Algorithm 5 An MH random walk algorithm to sample from $\pi(\beta, \kappa) = p(\beta, \kappa \,|\, y)$.

Initialize β and κ as $\beta^{(0)}$ and $\kappa^{(0)}$

for $t \leftarrow 1, \ldots, N$ **do**

$\beta^{(t)} \leftarrow \beta^{(t-1)}$

for $j \leftarrow 1, \ldots, m$ **do**

$\beta_j^{(t)} \leftarrow \text{MH\_RW\_}\beta(j, \beta^{(t)}, \kappa^{(t-1)}; \tau_{\beta j})$

end for

$\kappa^{(t)} \leftarrow \text{MH\_RW\_}\kappa(\beta^{(t)}, \kappa^{(t-1)}; \tau_\kappa)$

end for

distribution). Similarly, one might first update the posterior mean $M_\kappa^{(t)}$ and variance $V_\kappa^{(t)}$ for a few MCMC iterations before the first $\tau_\kappa^2 \leftarrow \lambda^2 V_\kappa^{(t)}$ assignment. In summary, first carry out few MCMC iterations without

updating the posterior mean and variance, then carry out few iterations updating them, but not changing τ_κ^2, and finally start to update τ_κ^2.

While Algorithm 6 gives steps for adjusting τ_κ when sampling κ using the MCMC Algorithm 3, similar adaptation can be used for any MCMC MH univariate random walk algorithm, including the MH normal random walk algorithm used to update any of the β_j's. The adaptive algorithm can also be extended to the multivariate random walk algorithm of Algorithm 4, and the details are given in Algorithm 7.

Algorithm 6 An adaptive step-size selection algorithm for MH normal random walk algorithm for κ that can be used in Algorithm 5.

1: Initialize κ as $\kappa^{(0)}$ and assume a fixed β
2: Initialize the proposal: set an initial value for τ_κ and let $\lambda \leftarrow 2.4$, $c \leftarrow 1$, $M_\kappa \leftarrow 0$, and $V_\kappa \leftarrow 0$
3: **for** $t \leftarrow 1, \ldots, N$ **do**
4: $\quad \kappa^{(t)} \leftarrow \text{MH\_RW\_}\kappa(\beta, \kappa^{(t-1)}; \tau_\kappa)$
5: $\quad M_\kappa^{(t)} \leftarrow M_\kappa^{(t-1)} + \frac{c}{t}(\kappa^{(t)} - M_\kappa^{(t-1)})$
6: $\quad V_\kappa^{(t)} \leftarrow V_\kappa^{(t-1)} + \frac{c}{t}((\kappa^{(t)} - M_\kappa^{(t)})^2 - V_\kappa^{(t-1)})$
7: $\quad \tau_\kappa^2 \leftarrow \lambda^2 V_\kappa^{(t)}$
8: **end for**

Algorithm 7 An adaptive *multivariate* MH normal random walk algorithm for the vector β that can be used in Algorithm 5.

1: Initialize the m-dimensional vector β as $\beta^{(0)}$) and assume a fixed value for κ
2: Initialize the proposal: Let $\lambda \leftarrow 2.4$ and $c \leftarrow 1$ and initialize the $m \times m$ proposal covariance matrix Σ, the m-dimensional sample mean vector $M(= 0)$, and the $m \times m$ sample covariance matrix $V(= 0)$
3: **for** $t \leftarrow 1, \ldots, N$ **do**
4: $\quad \beta^{(t)} \leftarrow \text{MH\_MRW\_}\beta(\beta^{(t-1)}, \kappa; \Sigma)$
5: $\quad M_\beta^{(t)} \leftarrow M_\beta^{(t-1)} + \frac{c}{t}(\beta^{(t)} - M_\beta^{(t-1)})$
6: $\quad V_\beta^{(t)} \leftarrow V_\beta^{(t-1)} + \frac{c}{t}((\beta^{(t)} - M_\beta^{(t)})(\beta^{(t)} - M_\beta^{(t)})^T - V_\beta^{(t-1)})$
7: $\quad \Sigma \leftarrow (\lambda/m)^2 V_\beta^{(t)}$
8: **end for**

Finally, there is an extension of the adaptive MCMC Algorithm 6 that also adapts the scaling parameter λ to yield a particular long-term targeted

acceptance rate. The details are given in Algorithm 8.

Algorithm 8 An adaptive step-size and scaling parameter selection algorithm for MH normal random walk algorithm for κ in Algorithm 6.

1: Initialize κ as $\kappa^{(0)}$ and assume a fixed β
2: Initialize the proposal: let $c \leftarrow 1$, $\hat{\alpha} \leftarrow 0.5$ (= to the target acceptance rate), $\lambda \leftarrow 2.4$, initialize τ_κ, and set the sample mean $M_\kappa = 0$ and the sample variance $V_\kappa = 0$
3: **for** $t \leftarrow 1, \ldots, N$ **do**
4: $\quad \kappa^{(t)} \leftarrow \text{MH\_RW\_}\kappa(\beta, \kappa^{(t-1)}; \tau_\kappa)$
5: $\quad M_\kappa^{(t)} \leftarrow M_\kappa^{(t-1)} + \frac{c}{t}(\kappa^{(t)} - M_\kappa^{(t-1)})$
6: $\quad V_\kappa^{(t)} \leftarrow V_\kappa^{(t-1)} + \frac{c}{t}((\kappa^{(t)} - M_\kappa^{(t)})^2 - V_\kappa^{(t-1)})$
7: $\quad \lambda \leftarrow \lambda \exp(\frac{c}{t}(\alpha^{(t)} - \hat{\alpha}))$, where $\alpha^{(t)} = \min(1, \alpha(\kappa^*, \kappa^{(t-1)}))$ is the acceptance rate of MH_RW_κ
8: $\quad \tau_\kappa^2 \leftarrow \lambda^2 V_\kappa^{(t)}$
9: **end for**

3 Exploratory Data Analysis

The seismic and air blast data come from a series of controlled explosion experiments called Humble Redwood I and II, performed at Kirtland Air Force Base near Alburquerque, NM (Foxall et al. (2010)). These experiments spanned a range of near-surface locations, including above and below ground explosions. For the most part, the yield was constant at 540 kg TNT. These data allow us to develop quantitative models of signal variation with yield, range, and HOB. The downside of the data is that these events are all from approximately the same location and hence observed by the same network of seismic and meteorological stations and thus only cover a very limited range of onditions. On the other hand, the fact that the events are very homogeneous (most of them with the same yield, but variation in HOB), make it easier to analyze the validity of the forward model.

We start with a short exploratory analysis in Figures 1–4. Figure 1 shows the raw scaled displacement plotted versus the scaled range, both on the log scale, for the HR events (using data from the BNZ channel). There is a (negative) linear relationship between the two quantities, which is indeed the relationship predicted by the displacement forward model in (1). There is an outlier corresponding to event HR2_5.

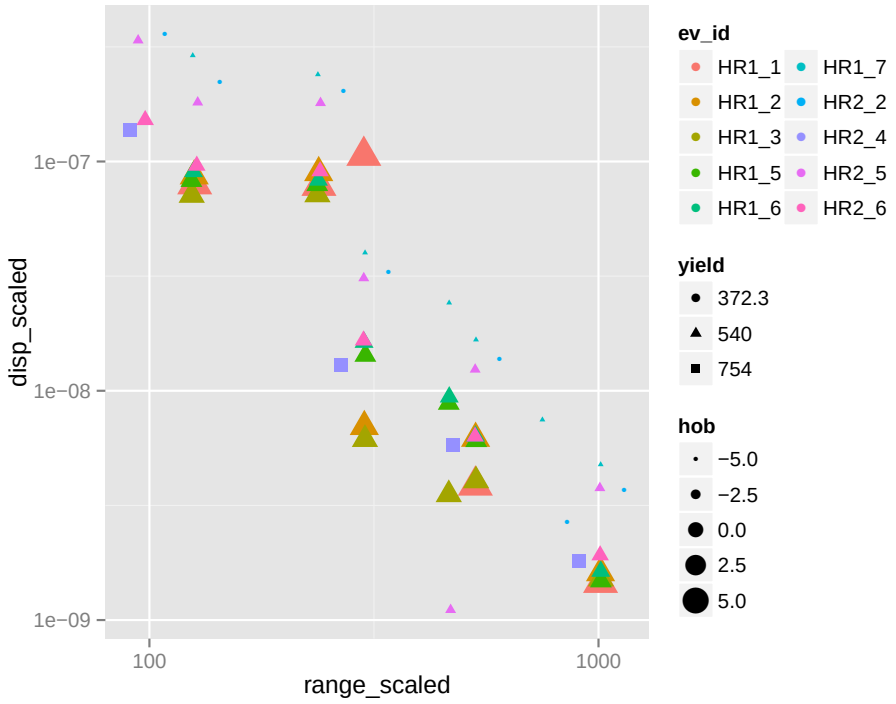

Figure 1: A plot of scaled displacement versus scaled range, both on the log scale, for HR events and BNZ channel, with different events marked by different colors.

Figure 2 shows a plot of the *residuals* of the log-transformed scaled displacement resulting from regressing it on the log-transformed scaled range. According to the forward model in (1), these residuals should follow a tanh relationship with the scaled HOB, which they appear to be in general, with several caveats. First, there is a clear shift in the relationship at scaled HOB just below 0. In addition, the distribution of residuals at any given scaled HOB is rather wide, and there are outliers (for example at scaled HOB = -0.4, corresponding to an observation at W1 station, which is the same outlier we observed in Figure 1).

The air pressure impulse analysis in Figures 3 and 4 agrees well with the proposed forward model in (3), with a generally linear relationship between log impulse and log scaled range, as seen in Figure 3, and a log relationship in the non-linear part of the model (albeit with a shift at scaled HOB below 0), as shown in Figure 4.

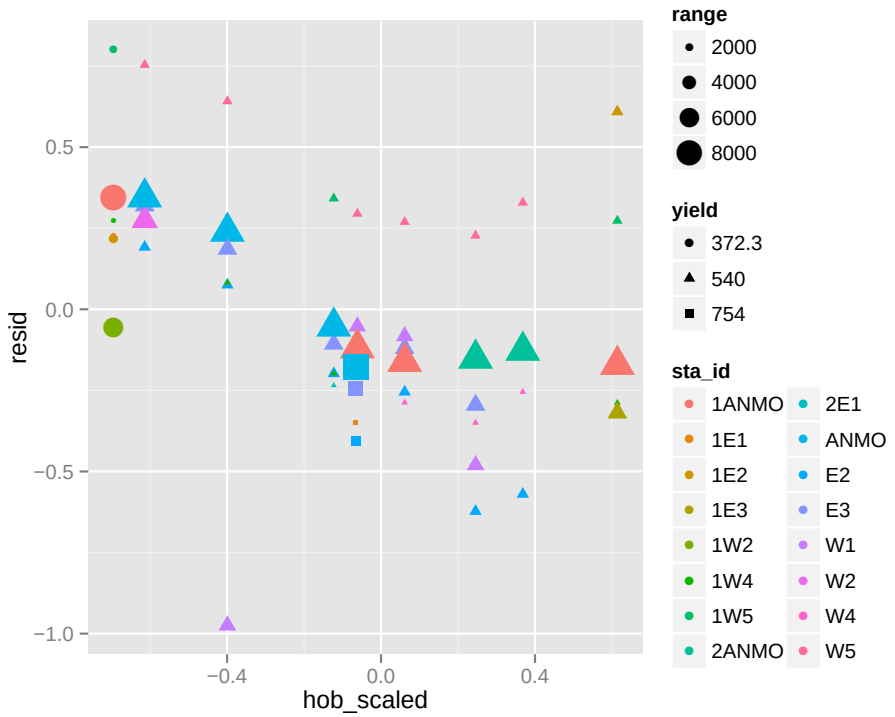

Figure 2: A plot of the residuals from the linear model $\log_{10}(d_{ij}/w_i^{(1/3)}) = \beta_{d1} + \beta_{d2}\log_{10}(r_{dij}/w_i^{(1/3)})$ versus scaled HOB for HR events and BNZ channel, with points colored by station ID. It shows that the residuals tend to follow the non-linear part of Model 2, $\beta_{d3}\tanh(\beta_{d4} + \beta_{d5}h_i^s)$, albeit with a shift at scaled HOB just below 0, with a great amount of noise around the tanh trend, as well as some outliers.

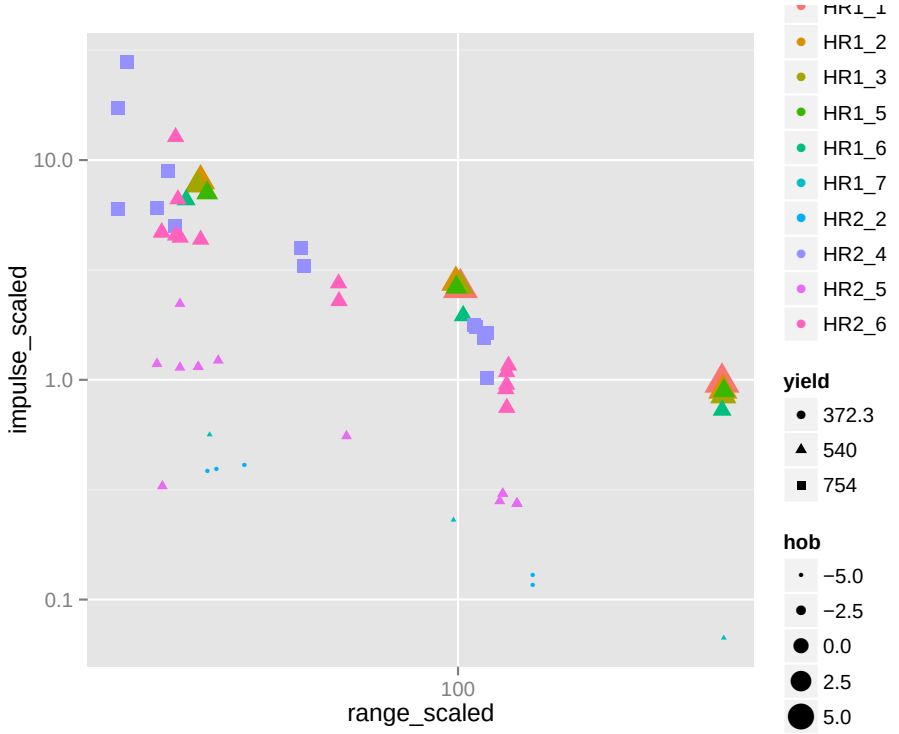

Figure 3: A plot of scaled air pressure impulse versus scaled range for HR events, both on the log scale.

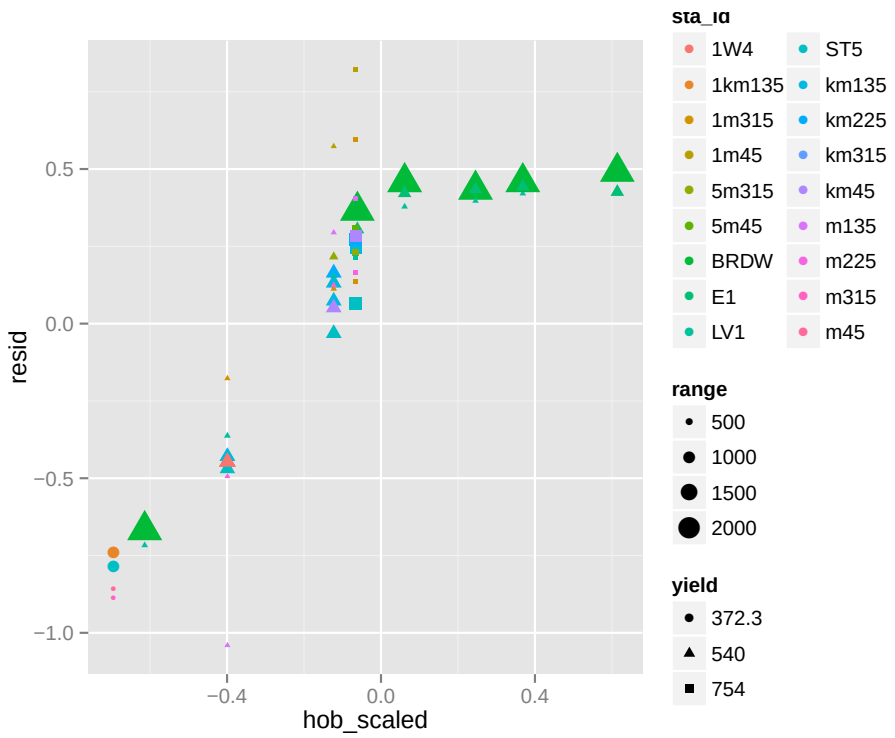

Figure 4: A plot of the residuals from the linear model $\log_{10}(a_{ij}/w_i^{(1/3)}) = \beta_{a1} + \beta_{a2}\log_{10}(r_{dij}/w_i^{(1/3)})$ versus scaled HOB for HR events, with points colored by station ID. It shows that the residuals tend to follow the non-linear part of Model 3, $\beta_{a3}h_i^s + \log_{10}(1 + 10^{10\beta_{a3}h_i^s})/10$, but with a shift at scaled HOB just below 0. Compared to Figure 2, the amount of noise around this expected relationship appears to be relatively small, and there fewer outliers, as well.

4 Bayesian Training of Seismic Displacement and Air Pressure Impulse Models

We will start with the basic relationships formulated in (2) between the seismic displacement and station range, HOB and yield, and similarly for air pressure impulse in (3). We assume that the yield (w_i) and HOB (h_i) are known for multiple events, and the main goal is to estimate the parameters of the models in (2) and (3). Within the Bayesian framework, this yields posterior realizations of all the unknown parameters of the training model, the β's plus other parameters that characterize the accuracy of the forward models. The trained forward models are then used in a Bayesian inversion for unknown yield and HOB of a new event, as will be outlined in the next Section.

4.1 Seismic Displacement Training Model

A simple starting point for a hierarchical Bayesian training model is given in Model 1, which trains a model to predict the scaled displacement, y^s_{dij}. The forward model is parameterized slightly differently from (2) in order to reduce "correlation" (or, equivalently, ensure identifiability) between parameters (in particular between β_{d3} and β_{d5}).

Model 1 A Simple hierarchical Bayesian training model for log10-transformed scaled displacement.

$$
\begin{aligned}
p(\log_{10} y^s_{dij} \,|\, \hat{y}^s_{dij}(\beta_d), \sigma_d) &= \mathrm{Gau}(\cdot \mid \log_{10} \hat{y}^s_{dij}(\beta_d), \sigma_d^2), \\
\log_{10} \hat{y}^s_{dij}(\beta_d) &= \beta_{d1} + \beta_{d2} \log_{10}(r^s_{dij}) \\
&\quad + \beta_{d3} \tanh(\beta_{d5}(h^s_i - \beta_{d4}))/\beta_{d5}, \\
p(\beta_{dk}) &\propto 1, \quad \text{for } k = 1, 2, 3 \text{ (non-informative)}, \\
p(\beta_{d4}) &= \mathrm{Gau}(\cdot \mid M_{d4} = 0, V_{d4} = 2^2), \\
p(\beta_{d5}) &= \mathrm{Gau}(\cdot \mid M_{d5} = 1, V_{d5} = 10^2)_{[0,\infty)} \\
p(1/\sigma_d^2) &\propto \mathrm{Gam}(\cdot \mid A_d = 10^{-3}, B_d = 10^{-6}), \qquad \text{(vague prior)}.
\end{aligned}
$$

Instead of training the model to predict the scaled displacement, the

model can be modified to train on the unscaled displacement using

$$\begin{aligned} p(\log_{10} y_{dij} \mid \hat{y}_{dij}(\beta_d), \sigma_d) &= \text{Gau}(\cdot \mid \log_{10} \hat{y}_{dij}(\beta_d), \sigma_d^2), \\ \log_{10} \hat{y}_{dij}(\beta_d) &= \beta_{d1} + \tfrac{1}{3}(1-\beta_{d2}) \log_{10}(w_i) + \beta_{d2} \log_{10}(r_{dij}) \\ &\quad + \beta_{d3} \tanh(\beta_{d5}(h_i / w_i^{1/3} - \beta_{d4}))/\beta_{d5}. \end{aligned} \tag{7}$$

In terms of posterior inference, the posterior distribution of $\theta_d = (\beta_{d1}, \ldots, \beta_{d5}, \sigma_d)$ will be identical to that derived using Model 1 with scaled displacement. Thus, it is just a matter of convenience which approach is taken.

4.1.1 Posterior Inference

Posterior inference can be carried out using MCMC. The forward model is relatively simple, with a linear component plus a non-linear part (in β_{d4} and β_{d5}). If β_{d4} and β_{d5} were fixed, then the model would be fully linear and a closed form solution would exist for the posterior (the Bayesian normal linear regression model). The Model 1 can be easily implemented in some of the more popular Bayesian MCMC inference engines, like STAN (`http://mc-stan.org`) and JAGS (`http://mcmc-jags.sourceforge.net`). These two samplers are fundamentally different and complement each other. STAN is a general purpose sampler that does not leverage any particular relationship between variables (e.g. like the conjugate normal-gamma relationship), but rather works with the whole joint log posterior PDF and samples from it jointly using adaptive Hamiltonian sampler. JAGS (just another Gibbs sampler, see Appendix A for more on the Gibbs samplers) builds a hierarchical (conditional) graph of the joint model and tries to use conjugate relationships between nodes (e.g., normal-normal, normal-gamma, etc.) when possible to sample each variable at a time and otherwise resorts to random walk Metropolis-Hastings samplers. Hence, STAN is often better at sampling highly correlated parameters, while JAGS can often result in an efficient sampler for a statistical model that consist of multiple conjugate blocks/nodes (that are uncorrelated or very weakly correlated). Both engines require the statistical model to be formulated in somewhat very similar language (that borrows from R and BUGS). STAN generates a C++ code that is then compiled and executed to generate realizations from the posterior. On the other hand, JAGS just processes the statistical model and produces an internal representation that is executed.

The STAN code for Model 1 is given in Code 1. The JAGS code is given in Code 2, which is slightly shorter as it does not require the data statements as it was used within R using the `rjags` pacakge (STAN was also used within R using the `rstan` package).

A pseudo-code to sample β's and $\kappa = 1/\sigma^2$ using Metropolis-Hastings normal random walk algorithm is very similar to the general algorithm outlined in Algorithm 5. The only difference would be the treatment of β_{d5}, which we assume is non-negative and would therefore use the same random walk proposal as the one used to sample $\kappa \geq 0$.

Code 1 The STAN code for Model 1.

```
## STAN model to train on seismic data (i.e., known yields and hob)
data {
    ## the seismic data:
    int<lower=0> n;                ## number of seismic displacement data
    vector[n] log10_disp_scaled;   ## = log10(displacement/yield^(1/3))
    vector[n] log10_range_scaled;  ## = log10(range/yield^(1/3))
    vector[n] hob_scaled;          ## = hob/yield^(1/3)
    vector<lower=0>[n] stdev;   ## = estimated stdev of disp obs
    int<lower=0> n_e;           ## = # of events
    int<lower=0> i_e[n];        ## index to event/explosion, between 1 and n_e
    int<lower=0> n_s;           ## = # of stations
    int<lower=0> i_s[n];        ## index to station, between 1 and n_s
}

parameters {
    vector[5] beta;            ## the param of the seismic model
    real<lower=0> kappa;       ## the precision = 1/variance
}

transformed parameters {
    real<lower=0> sigma;
    vector[n] mu;
    sigma <- 1.0/sqrt(kappa);
    for(i in 1:n) {
        mu[i] <- beta[1] + beta[2]*log10_range_scaled[i] +
            beta[3]*tanh(beta[5]*(hob_scaled[i]-beta[4]))/beta[5];
    }
}

model {
    log10_disp_scaled ~ normal(mu, sigma);
    beta[4] ~ normal(0,2);  ## somewhat informative
    beta[5] ~ normal(1,10) T[0,];  ## pretty vague
    kappa ~ gamma(1E-3,1E-6);  ## pretty vague
}
```

With the STAN code, it took roughly 17 seconds to yield the C++ executable code and then another 7.7 seconds to run 4 MCMC chains for 5000 iterations (retaining only every 5th iteration of the last 2500 iterations),

Code 2 The JAGS code for Model 1.

```
## JAGS model to train on seismic data (i.e., known yield and hob)
model {
    for(i in 1:n) {
        mu[i] <- beta[1] + beta[2]*log10_range_scaled[i] +
            beta[3]*tanh(beta[5]*(hob_scaled[i] - beta[4]))/beta[5]
        log10_disp_scaled[i] ~ dnorm(mu[i], kappa)
    }
    for(j in 1:3)
        beta[j] ~ dnorm(0,1E-10)  ## "flat" prior
    beta[4] ~ dnorm(0,1/(2*2))  ## somewhat informative prior
    beta[5] ~ dnorm(1,1/(20*20)) T(0,)
    sigma <- 1.0/sqrt(kappa)
    kappa ~ dgamma(1E-3,1E-6)   ## vague gamma prior
}
```

using the HR events (and channel BNZ) and removing two outliers. A posterior summary is given in Table 1 and Figures 5–11 provide graphical summaries. For the graphical summaries we note the following:

- The MCMC chains are well mixed (Figure 5).
- There is a limited correlation between the parameters (β's and σ), except between β_{d1} and β_{d2} and between β_{d3} and β_{5d}, as expected (although the correlation between β_{d1} and β_{d2} can be removed by centering $\log_{10}(r^s_{dij})$).
- There is an obvious station bias in the residuals, particularly for the `W5` station.

```
Inference for Stan model: seismic_training_model_1.
4 chains, each with iter=5000; warmup=2500; thin=5;
post-warmup draws per chain=500, total post-warmup draws=2000.

         mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
beta[1] -2.42    0.01 0.28 -2.95 -2.61 -2.42 -2.24 -1.88  1863    1
beta[2] -2.04    0.00 0.11 -2.25 -2.11 -2.05 -1.97 -1.84  1889    1
beta[3] -2.91    0.04 1.51 -6.42 -3.78 -2.61 -1.77 -0.87  1381    1
beta[4] -0.30    0.00 0.10 -0.52 -0.37 -0.30 -0.22 -0.14  1392    1
beta[5] 12.78    0.18 6.70  2.77  7.74 11.69 16.98 28.16  1408    1
sigma    0.26    0.00 0.02  0.21  0.24  0.25  0.27  0.31  1857    1

Samples were drawn using NUTS(diag_e) at Thu Oct 23 11:51:50 2014.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

Table 1: Posterior summary statistics for the seismic training Model 1 from STAN.

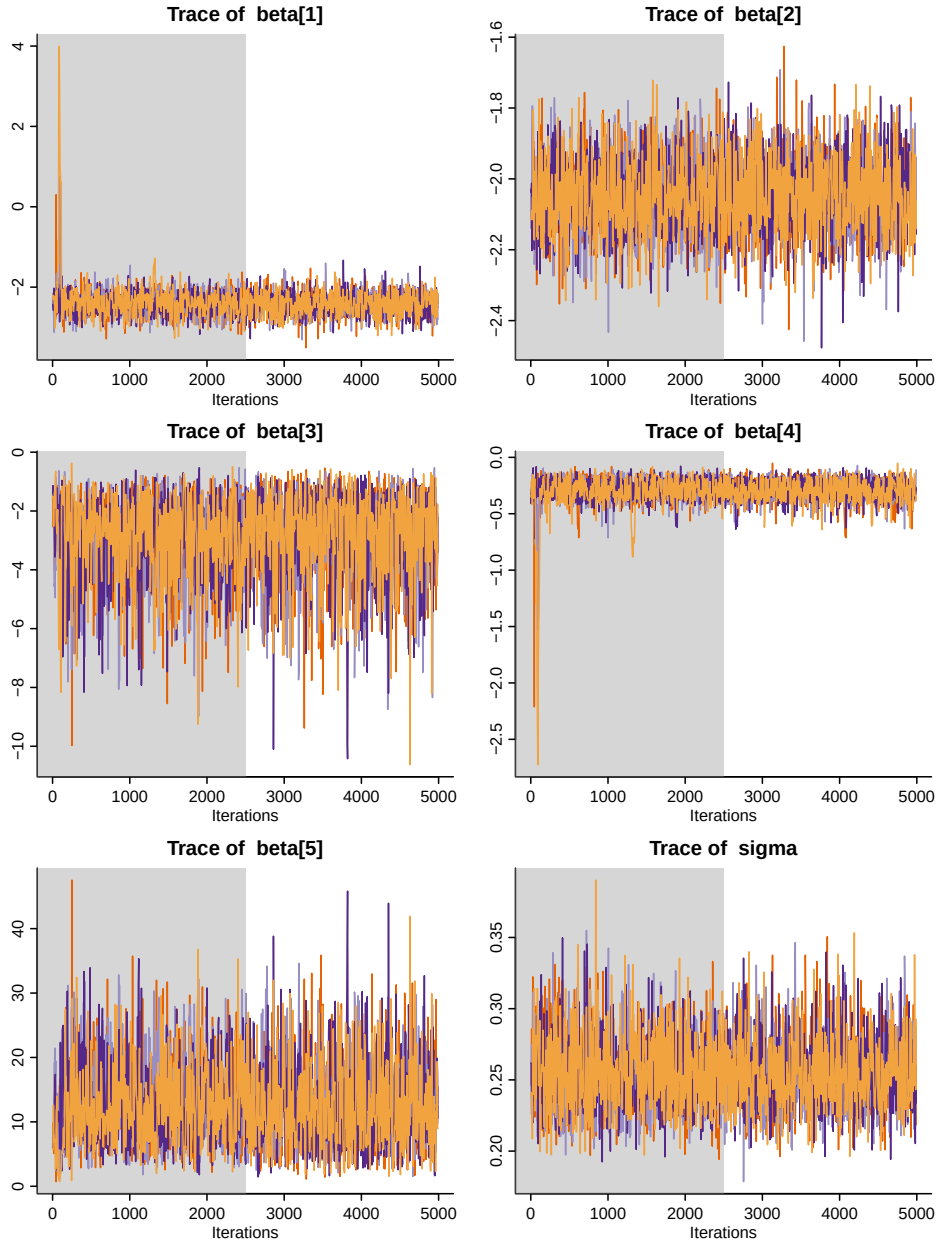

Figure 5: MCMC trace plots for the variables in Model 1 from STAN Code 1.

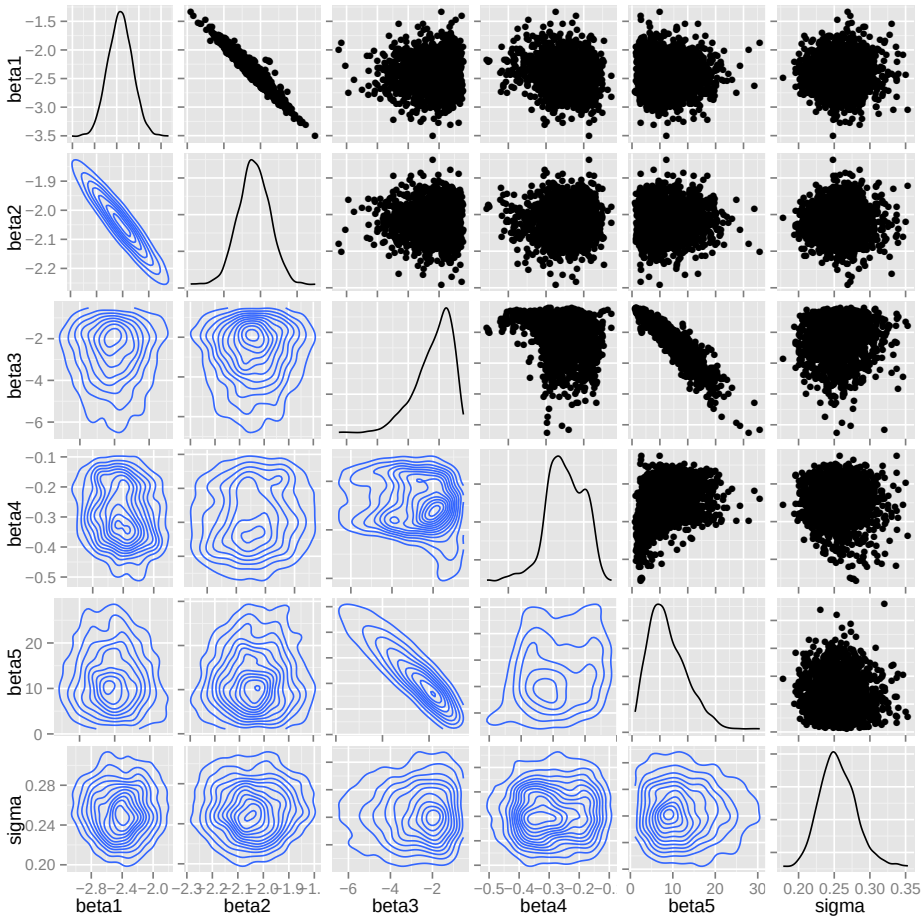

Figure 6: Graphical summary of the posterior distributions of $\beta_{d1}, \ldots, \beta_{d5}$ and σ in Model 1 from STAN Code 1.

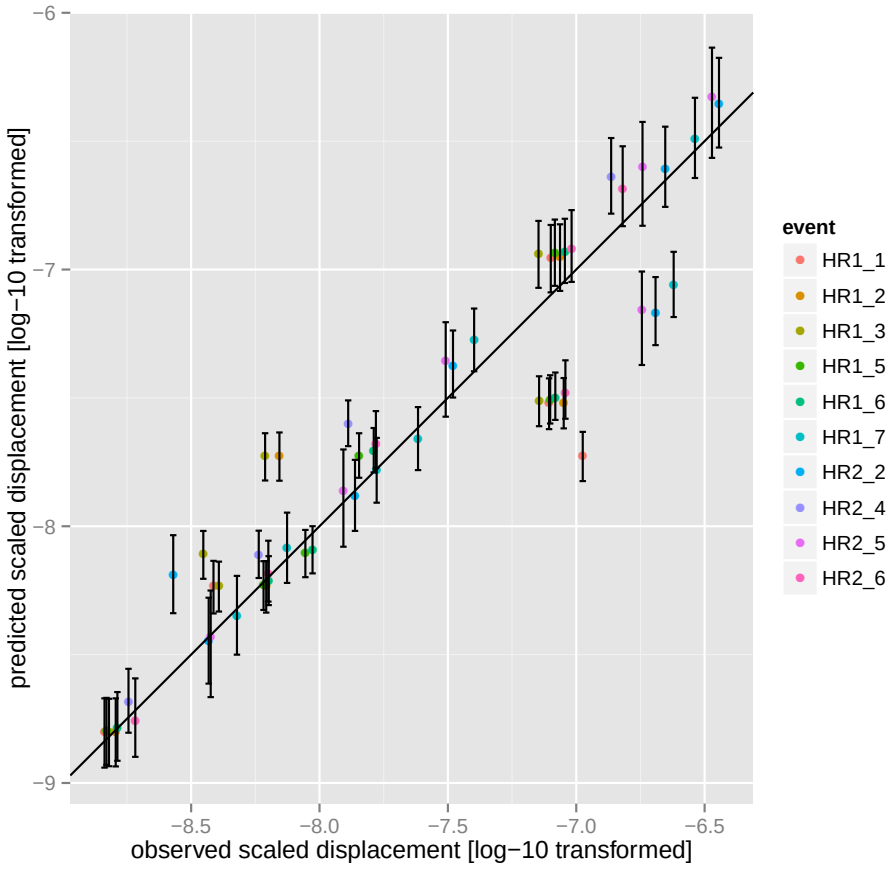

Figure 7: The posterior mean predicted values of $\log_{10} \hat{y}_{dij}(\beta_d)$ along with empirical 95% prediction intervals for Model 1 from STAN Code 1.

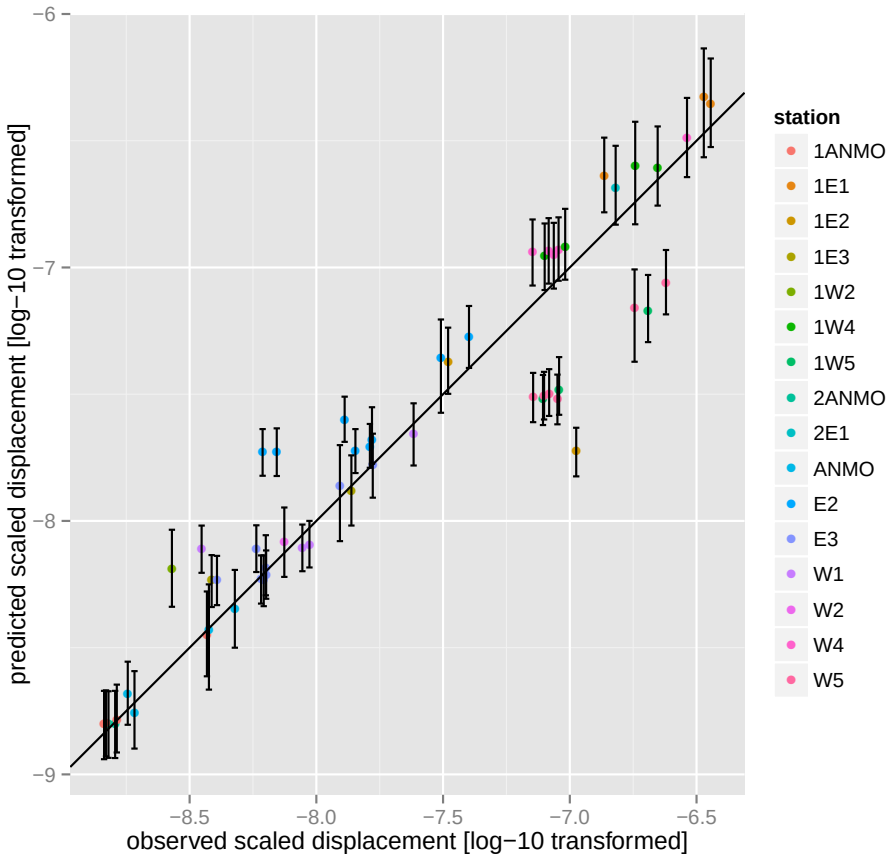

Figure 8: Same as Figure 7, except with the prediction points colored by station ID (rather than event ID).

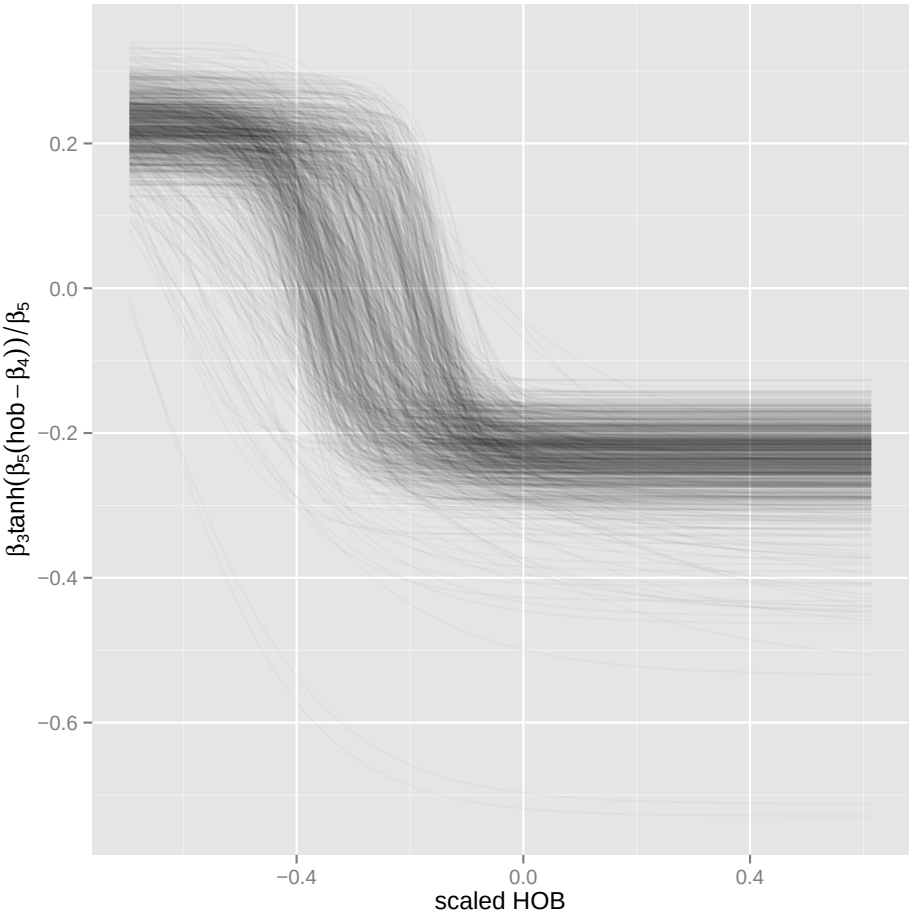

Figure 9: One thousand posterior realizations of the non-linear part in the forward model, $\beta_{d3}\tanh(\beta_{d5}(h_i^s - \beta_{d4}))/\beta_{d5}$, in Model 1 from STAN Code 1.

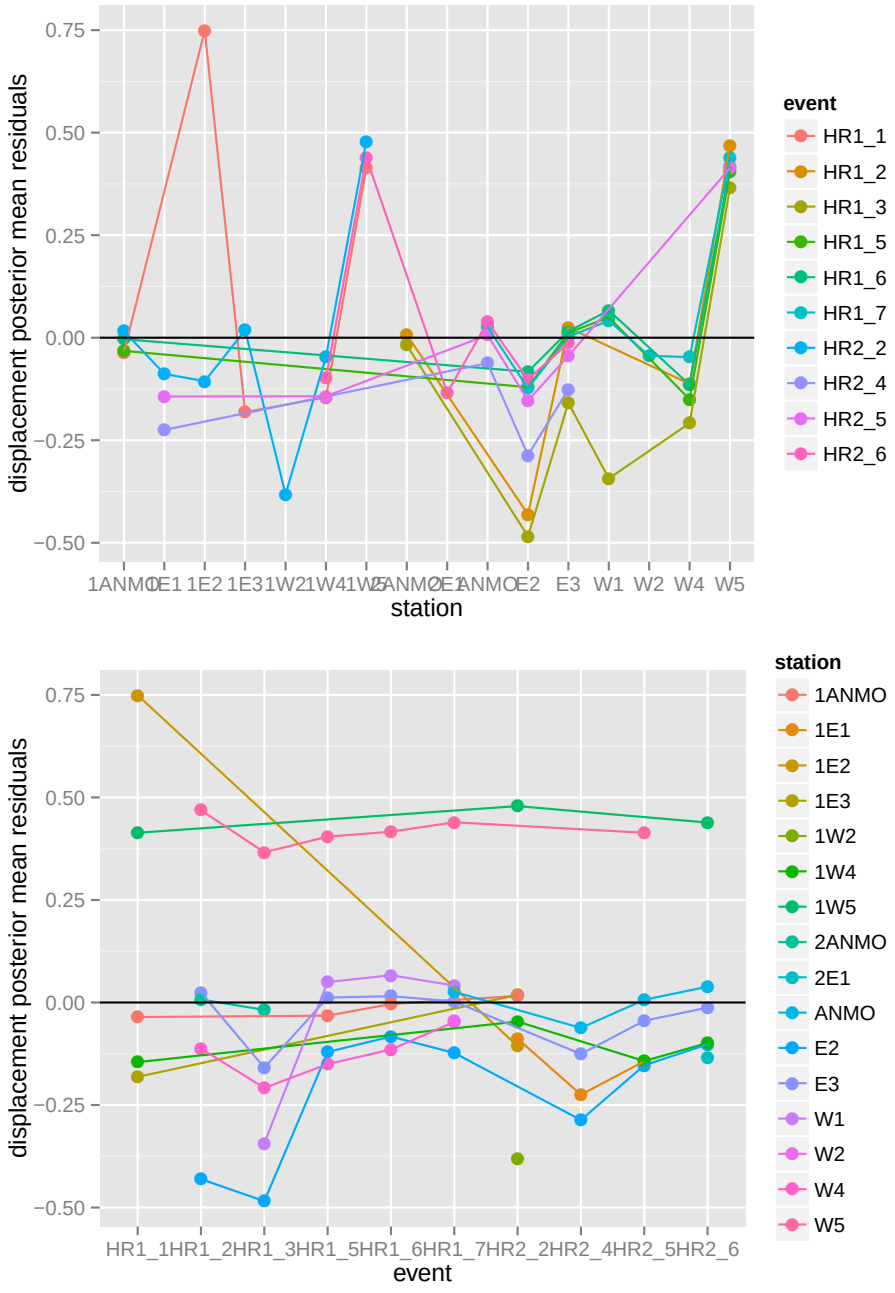

Figure 10: The posterior mean residuals versus (top) seismic station IDs and (bottom) explosion/event IDs from Model 1 from STAN Code 1.

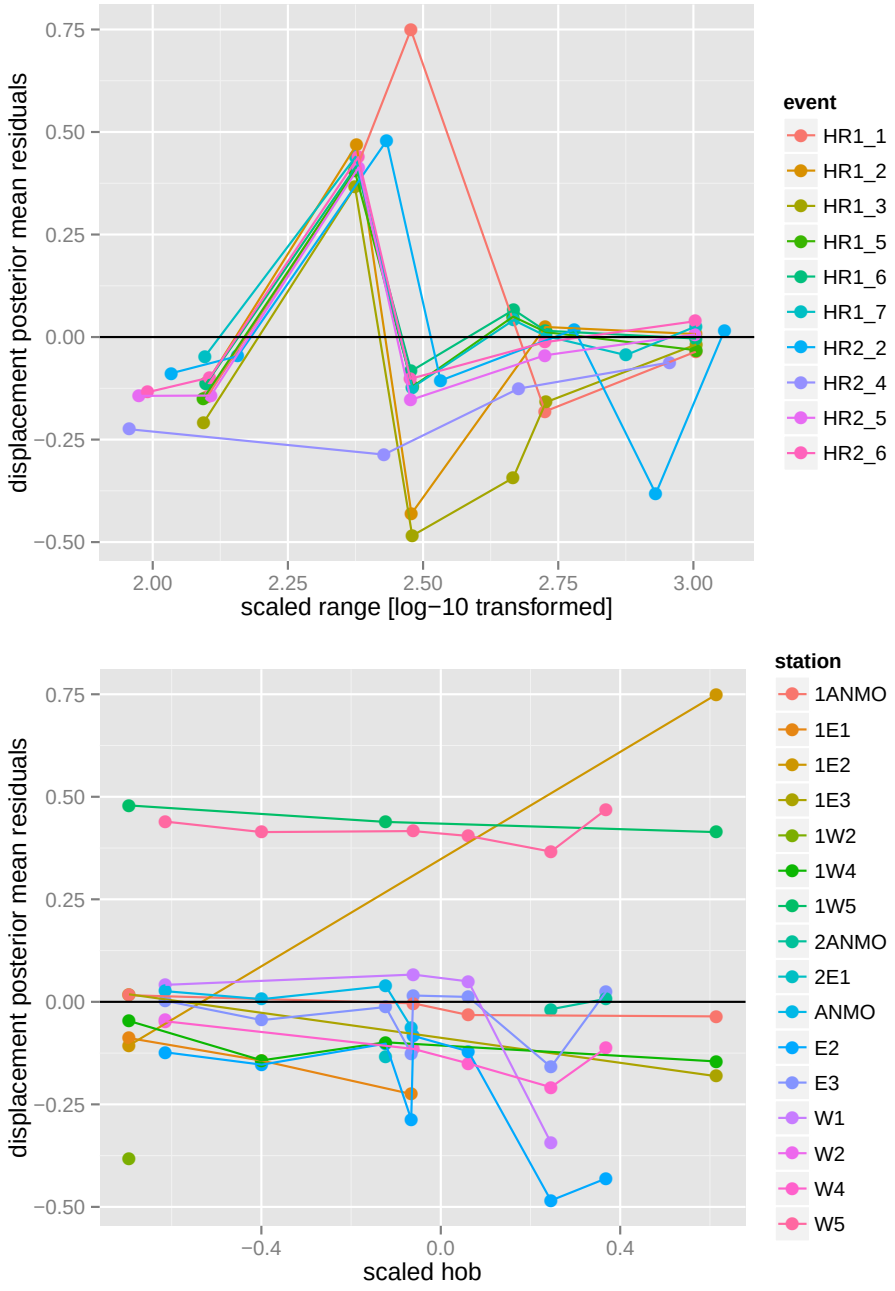

Figure 11: Similar to Figure 10, but with the posterior mean residuals versus (top) the scaled range and (bottom) scaled HOB from Model 1 from STAN Code 1.

The posterior inference from the JAGS code, Code 2, is very similar to the STAN inference. It took roughly 0.7 seconds for 4 chains to run for 2500 iterations in the "burn-in" phase and then another 1.6 seconds to run the 4 chains for additional 2500 iterations, storing every 5th iteration. The posterior summary statistics are in Table 2 while the MCMC trace plots are in Figure 12. A graphical summary of the univariate and bivariate posteriors is provided in Figure 13.

JAGS has little more trouble sampling correlated parameters, as expected, while it runs faster (hence, one could run JAGS for more iterations to yield longer Markov chains and compensate for higher auto-correlation in the posterior sample).

	Mean	SD	Naive SE	Time-series SE
beta[1]	-2.4670429	0.26891723	0.0060131721	0.0300094757
beta[2]	-2.0267436	0.10221778	0.0022856590	0.0116044134
beta[3]	-2.9762836	1.49984526	0.0335375596	0.0781160491
beta[4]	-0.2960836	0.10178763	0.0022760406	0.0042387262
beta[5]	13.0838635	6.72050013	0.1502749513	0.3873460576
sigma	0.2536253	0.02460132	0.0005501022	0.0005499048

	2.5%	25%	50%	75%	97.5%
beta[1]	-3.0256149	-2.6366760	-2.4620398	-2.2985963	-1.9208534
beta[2]	-2.2261135	-2.0932870	-2.0295378	-1.9613682	-1.8103676
beta[3]	-6.3991524	-3.7829280	-2.6861625	-1.8285224	-0.8880836
beta[4]	-0.5042731	-0.3626564	-0.2953452	-0.2184952	-0.1326772
beta[5]	3.0956481	7.9998005	12.0819803	17.0741883	28.7803350
sigma	0.2102470	0.2362853	0.2522070	0.2688024	0.3070868

Table 2: Posterior summary statistics for the seismic training Model 1 from JAGS.

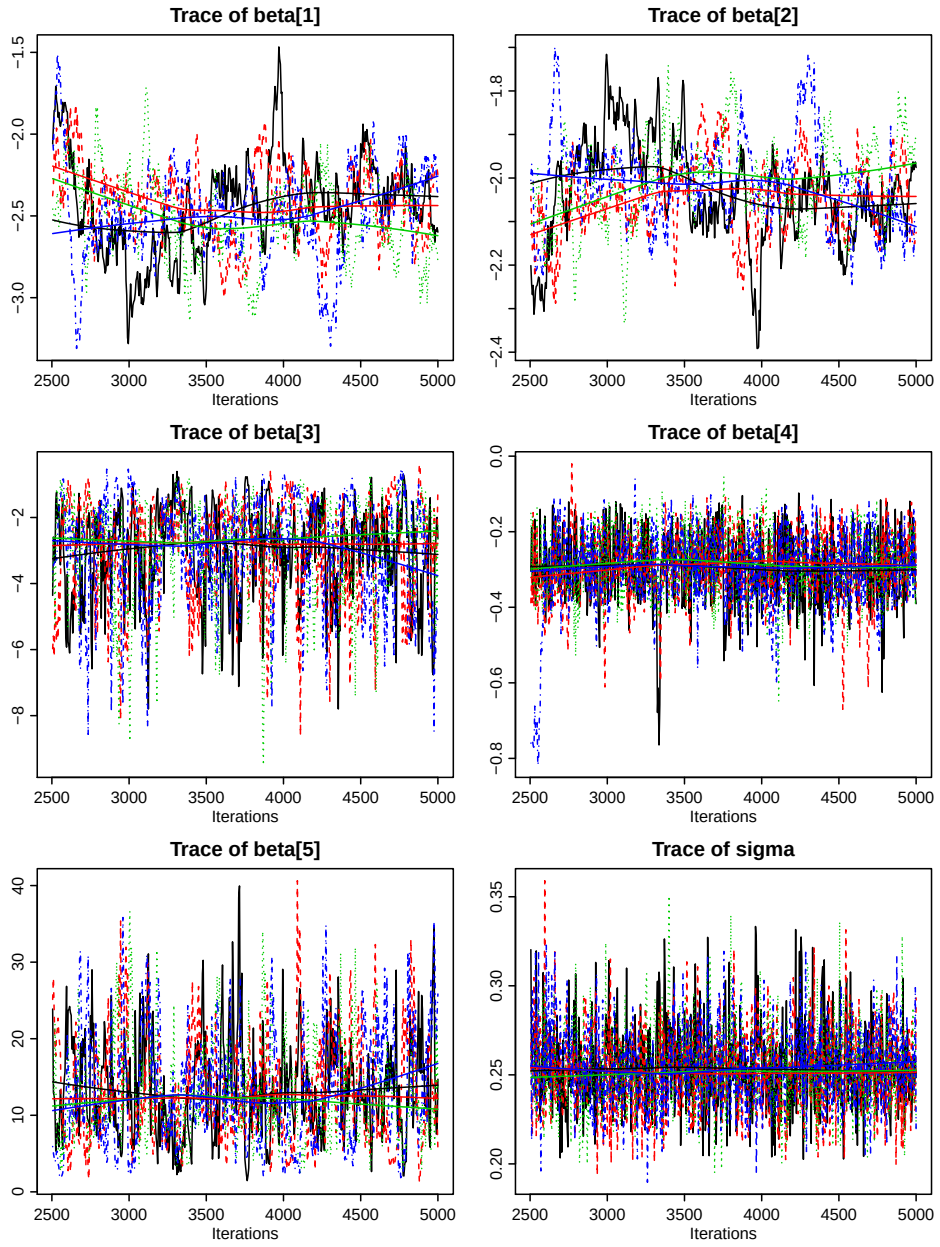

Figure 12: MCMC trace plots for the variables in Model 1 from JAGS Code 2.

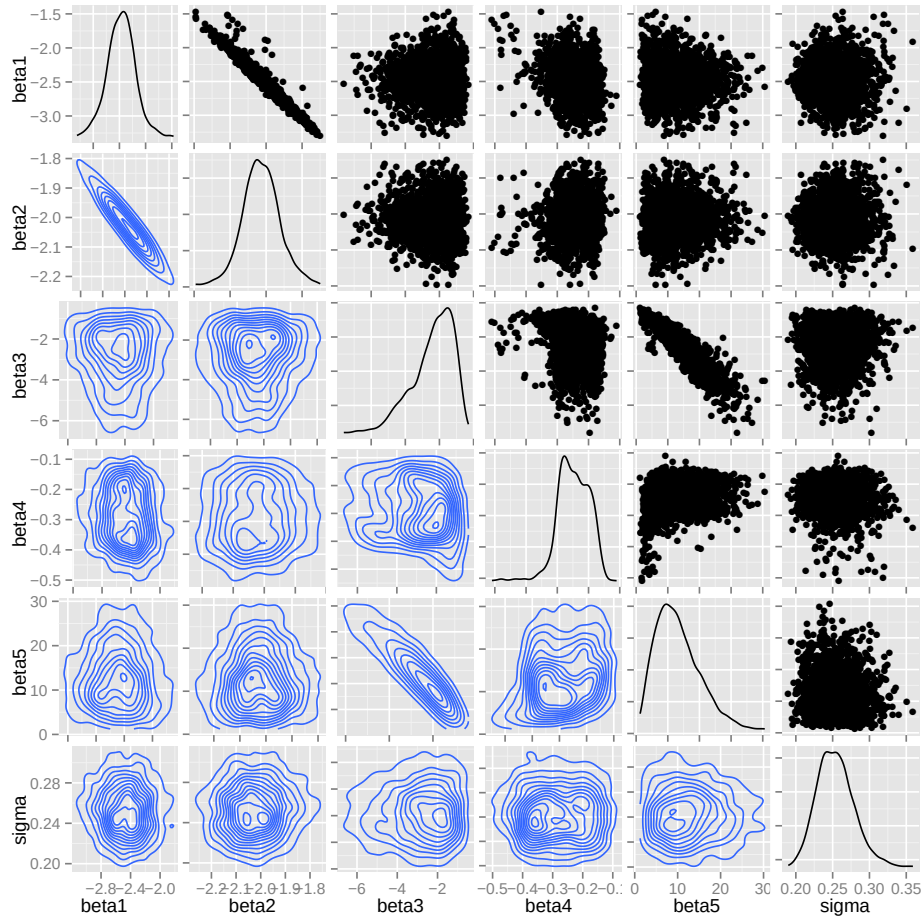

Figure 13: Graphical summary of the posterior distributions of $\beta_{d1}, \ldots, \beta_{d5}$ and σ in Model 1 from JAGS Code 2.

4.1.2 Extension

From the previous posterior analysis we notice an obvious station-specific bias in the posterior residuals. A natural extension to Model 1 is to add station-specific random effects to the intercept (β_{d1}), as outlined in Model 2. Note that in this formulation the intercept of the forward model is given by $\beta_{d1} + b_{d1,j}$ where the station-specific deviations, $b_{d1,j}$'s, are assumed to be normally distributed with mean zero and variance σ^2_{bd1}, which is treated as unknown (and estimated). That is, the population of the station-specific deviations from the overall intercept is apriori assumed to have a normal distribution. There is a "soft" lack of identifiability in the model as one can add a small value Δ to the global intercept β_{d1} and subtract the same Δ from each of $b_{d1,j}$'s without changing the prediction. Therefore, in addition to just looking at the posterior PDF of each $b_{d1,j}$, it is helpful to look at the posterior PDF of the actual intercept, $\beta_{d1} + b_{d1,j}$.

Model 2 An extension of Model 1 for the seismic displacement data that introduces station-specific random effects, $b_{d1,j}$'s, for the intercept.

$$
\begin{aligned}
p(\log_{10} y^s_{dij} \,|\, \hat{y}^s_{dij}(\beta_d), \sigma_d) &= \mathrm{Gau}(\cdot \mid \log_{10} \hat{y}^s_{dij}(\beta_d), \sigma^2_d), \\
\log_{10} \hat{y}^s_{dij}(\beta_d) &= \beta_{d1} + b_{d1,j} + \beta_{d2} \log_{10}(r^s_{dij}) \\
&\qquad + \beta_{d3} \tanh(\beta_{d5}(h^s_i - \beta_{d4}))/\beta_{d5}, \\
p(\beta_{dk}) &\propto 1, \quad \text{for } k = 1, 2, 3 \text{ (non-informative)}, \\
p(\beta_{d4}) &= \mathrm{Gau}(\cdot \,|\, M_{d4} = 0, V_{d4} = 2^2), \\
p(\beta_{d5}) &= \mathrm{Gau}(\cdot \,|\, M_{d5} = 1, V_{d5} = 10^2)_{[0,\infty)} \\
p(b_{d1,j}) &= \mathrm{Gau}(0, \sigma^2_{bd1}), \quad \text{for } j = 1, \ldots, n_d \text{ (random-effects)}, \\
p(1/\sigma^2_d) &= \mathrm{Gam}(\cdot \mid A_d = 10^{-3}, B_d = 10^{-6}), \qquad \text{(vague prior)}, \\
p(1/\sigma^2_{bd1}) &= \mathrm{Gam}(\cdot \mid A_{db1} = 10^{-3}, B_{db1} = 10^{-6}), \qquad \text{(vague prior)}.
\end{aligned}
$$

The MCMC sampler for Model 1 can be extended to include MH steps to sample $b_{d1,j}$'s individually and then σ^2_{db1}. Algorithm 9 outlines an MH algorithm to sample $b_j = b_{d1,j}$, with all the other parameters held fixed at the current iteration value. It mirrors Algorithm 2 to sample any given β_k and uses the general notation outlined there, in particular using precisions instead of variances; $\kappa = 1/\sigma^2$ and $\kappa_b = 1/\sigma^2_{bd1}$ in Algorithm 9. Similarly, the MH algorithm to sample $\kappa_b = 1/\sigma^2_{bd1}$ is given in Algorithm 10, and it uses the same notation as that used in Algorithm 3 for sampling $\kappa = 1/\sigma^2$.

Algorithm 9 An MH random walk algorithm for a random effect $b_j = b_{d1,j}$.

procedure MH_RW_$b(j,\ b^{(t)},\ \beta^{(t)},\ \kappa^{(t)},\ \kappa_b^{(t)};\ \tau)$
  Let $b^* \leftarrow b^{(t)} = (b_1^{(t)}, \ldots, b_m^{(t)})$
  Only change the j-th random-effect using random-walk, $b_j^* \leftarrow b_j^{(t)} + \epsilon$, where $\epsilon \sim \text{Gau}(0, \tau^2)$, with τ provided (the "step-size")
  Compute $\hat{y}(., b^*) = \hat{y}(\beta^{(t)}, b^*)$, reflecting the change in b_j^*
  Compute the log-acceptance rate,

$$\begin{aligned}
\log \alpha(b_j^*, b_j^{(t)}) &= \sum_{i=1}^{n} \left(\log \text{Gau}(y_i \,|\, \hat{y}_i(., b^*), 1/\kappa^{(t)}) - \log \text{Gau}(y_i \,|\, \hat{y}_i(., b^{(t)}), 1/\kappa^{(t)}) \right) \\
&\quad + \left(\log \text{Gau}(b_j^* \,|\, 0, 1/\kappa_b^{(t)}) - \log \text{Gau}(b_j^{(t)} \,|\, 0, 1/\kappa_b^{(t)}) \right) \\
&= -\frac{\kappa^{(t)}}{2} \sum_{i=1}^{n} \left((y_i - \hat{y}_i(., b^*))^2 - (y_i - \hat{y}_i(., b^{(t)}))^2 \right) \\
&\quad - \frac{\kappa_b^{(t)}}{2} \left((b_j^*)^2 - (b_j^{(t)})^2 \right)
\end{aligned}$$

  Generate $u \sim \text{Unif}(0, 1)$
  if $\alpha(b_j^*, b_j^{(t)}) \geq u$ **then** $b_j^{(t+1)} \leftarrow b_j^*$ **else** $b_j^{(t+1)} \leftarrow b_j^{(t)}$
  return $b_j^{(t+1)}$
end procedure

A single MCMC iteration that samples all the parameters of Model 2 might then consist of the following steps:

1. Loop through β_k's and update each using an MH algorithm similar to the one in Algorithm 2.
2. Sample $\sigma^2 = 1/\kappa$ using an MH algorithm similar to the one in Algorithm 3.
3. Loop through b_j's and update each using an MH algorithm similar to the one in Algorithm 9.
4. Sample $\sigma_{bd1}^2 = 1/\kappa_b$ using an MH algorithm similar to the one in Algorithm 10.

The above can be easily extended to sample any additional random effects introduced for other β parameters of Model 2.

Algorithm 10 An MH random walk algorithm for $\kappa_b = 1/\sigma_b^2$, where σ_b^2 is the variance of the random effects $b_1, \ldots, b_m$.

procedure MH_RW_κ_b($b^{(t)}$, $\kappa_b^{(t)}$; τ)

Let $\kappa_b^* \leftarrow \kappa_b^{(t)} + \epsilon$, where $\epsilon \sim \text{Gau}(0, \tau^2)$. If $\kappa_b^* < 0$, then $\kappa_b^* \leftarrow -\kappa_b^*$

Compute the log-acceptance rate,

$$\begin{aligned}\log \alpha(\kappa_b^*, \kappa_b^{(t)}) &= \sum_{j=1}^{m} \Big(\log \text{Gau}(b_j^{(t)} \mid 0, 1/\kappa_b^*) - \log \text{Gau}(b_j^{(t)} \mid 0, 1/\kappa_b^{(t)}) \Big) \\ &\quad + \Big(\log \text{Gam}(\kappa_b^* \mid A_b, B_b) - \log \text{Gam}(\kappa_b^{(t)} \mid A_b, B_b) \Big) \\ &= (m/2 + A_b - 1)(\log \kappa_b^* - \log \kappa_b^{(t)}) - (mS^2/2 + B_b)(\kappa_b^* - \kappa_b^{(t)})\end{aligned}$$

where $S = \sqrt{\frac{1}{m} \sum_{j=1}^{m} (b_j^{(t)})^2}$

Generate $u \sim \text{Unif}(0, 1)$

if $\alpha(\kappa_b^*, \kappa_b^{(t)}) \geq u$ **then** $\kappa_b^{(t+1)} \leftarrow \kappa_b^*$ **else** $\kappa_b^{(t+1)} \leftarrow \kappa_b^{(t)}$

return $\kappa_b^{(t+1)}$

end procedure

The STAN code in Code 1, corresponding to Model 1, can be easily updated to sample from Model 2 and is provided in Code 3. A posterior summary of 4 MCMC chains, using the last 2500 iterations out of 5000, is provided in Table 3 for β's and σ. A graphical summary of the station-specific intercept random effects, $b_{d1,j}$'s, is provided in Figure 14 and summary of the predicted vs observed values is given in Figures 15 and 16, while Figures 17 and 17 summarize the posterior residuals.

In summary,

- The standard deviation of the residuals (σ) is smaller than in the previous model: its posterior mean is 0.14 versus 0.28 in the model without the station-specific random effects.
- Many of the station-specific biases that were significantly different from 0 have been successfully removed by modeling the station-specific random effects.
- There is one obvious outlier station, 1E2, that only has two observations.

Code 3 STAN code for Model 2.

```
## STAN model to train on seismic data (i.e., known yields and hob)
data {
    ## the seismic data:
    int<lower=0> n;                ## number of seismic displacement data
    vector[n] log10_disp_scaled;   ## = log10(displacement/yield^(1/3))
    vector[n] log10_range_scaled;  ## = log10(range/yield^(1/3))
    vector[n] hob_scaled;          ## = hob/yield^(1/3)
    vector<lower=0>[n] stdev;   ## = estimated stdev of disp obs
    int<lower=0> n_e;           ## = # of events
    int<lower=0> i_e[n];        ## index to event/explosion, between 1 and n_e
    int<lower=0> n_s;           ## = # of stations
    int<lower=0> i_s[n];        ## index to station, between 1 and n_s
}

parameters {
    vector[5] beta;            ## the param of the seismic model
    real<lower=0> kappa;       ## the precision = 1/variance
    vector[n_s] b1_sta;           ## station random effects
    real<lower=0> kappa_b1_sta;   ## the precision of the sta random effects
}

transformed parameters {
    real<lower=0> sigma;
    real<lower=0> sigma_b1_sta;
    vector[n] mu;
    sigma <- 1.0/sqrt(kappa);
    sigma_b1_sta <- 1.0/sqrt(kappa_b1_sta);
    for(i in 1:n) {
        mu[i] <- (beta[1] + b1_sta[i_s[i]]) + beta[2]*log10_range_scaled[i] +
            beta[3]*tanh(beta[5]*(hob_scaled[i]-beta[4]))/beta[5];
    }
}

model {
    log10_disp_scaled ~ normal(mu, sigma);
    beta[4] ~ normal(0,2);  ## somewhat informative
    beta[5] ~ normal(1,10) T[0,];  ## pretty vague
    kappa ~ gamma(1E-3,1E-6);  ## pretty vague
    b1_sta ~ normal(0,sigma_b1_sta);
    kappa_b1_sta ~ gamma(1E-3,1E-6);
}
```

- At the same time, including the station-specific random effects may not be feasible in some cases because one may not always have repeated data at a given station to estimate that station's random effect.

```
Inference for Stan model: seismic_training_model_2.
4 chains, each with iter=5000; warmup=2500; thin=5;
post-warmup draws per chain=500, total post-warmup draws=2000.

         mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
beta[1] -2.41    0.01 0.44 -3.26 -2.68 -2.39 -2.13 -1.53  1350    1
beta[2] -2.06    0.00 0.17 -2.39 -2.16 -2.06 -1.95 -1.72  1380    1
beta[3] -2.59    0.03 1.34 -5.98 -3.33 -2.25 -1.55 -0.96  1602    1
beta[4] -0.26    0.00 0.08 -0.39 -0.32 -0.26 -0.20 -0.14  1736    1
beta[5] 11.10    0.15 6.05  3.38  6.38  9.82 14.33 26.23  1575    1
sigma    0.14    0.00 0.02  0.11  0.13  0.14  0.15  0.18  1706    1

Samples were drawn using NUTS(diag_e) at Thu Oct 23 21:31:28 2014.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

Table 3: Posterior summary statistics for the seismic training model outlined in Code 3 from STAN.

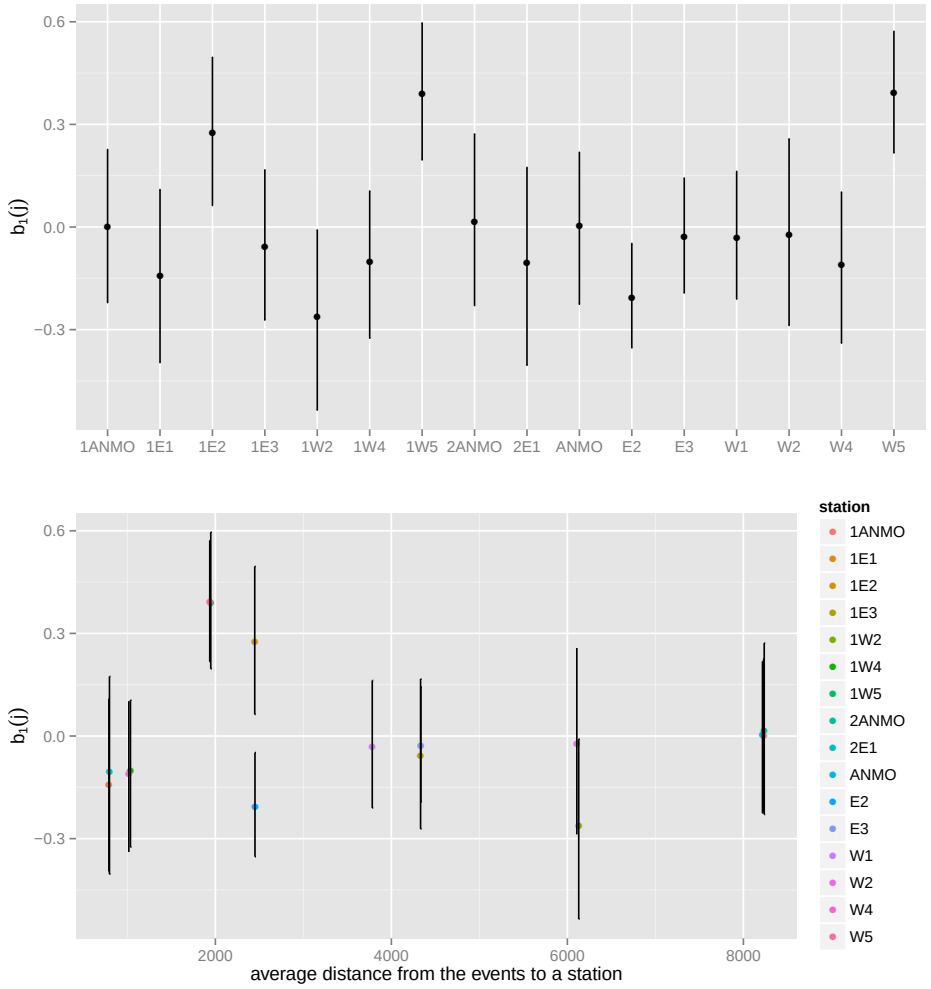

Figure 14: The posterior mean predicted values of the station-specific random effects for the intercept, $b_{d1,j}$'s, along with empirical 95% prediction intervals, for Model 2 from STAN Code 3.

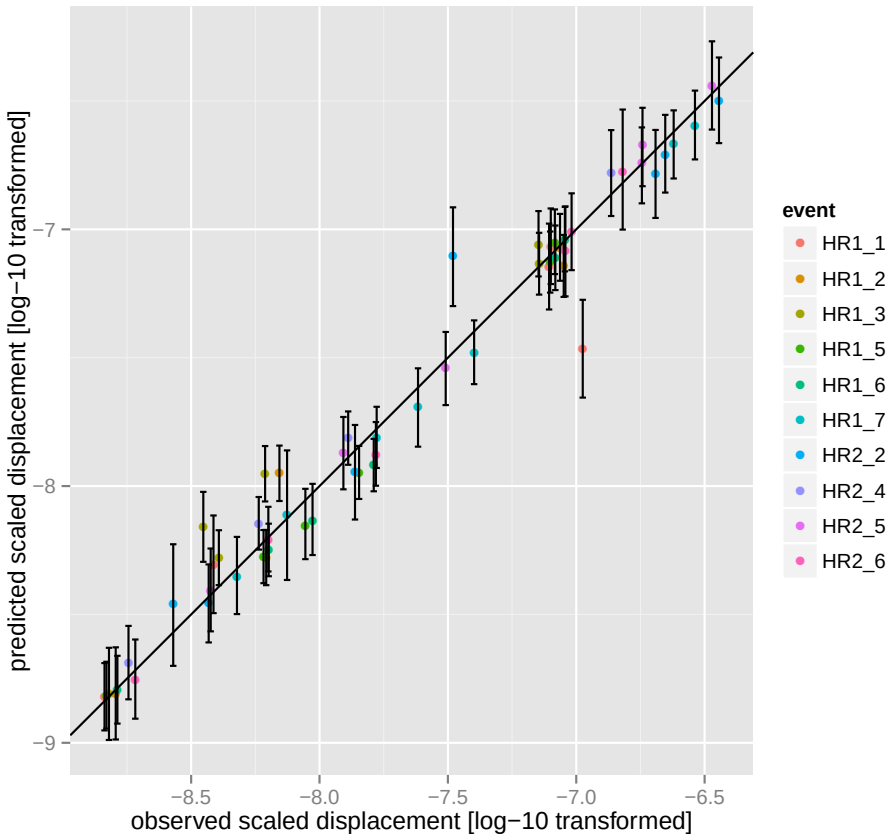

Figure 15: The posterior mean predicted values of $\log_{10} \hat{y}_{dij}(\beta_d)$, along with empirical 95% prediction intervals, for Model 2 from STAN Code 3.

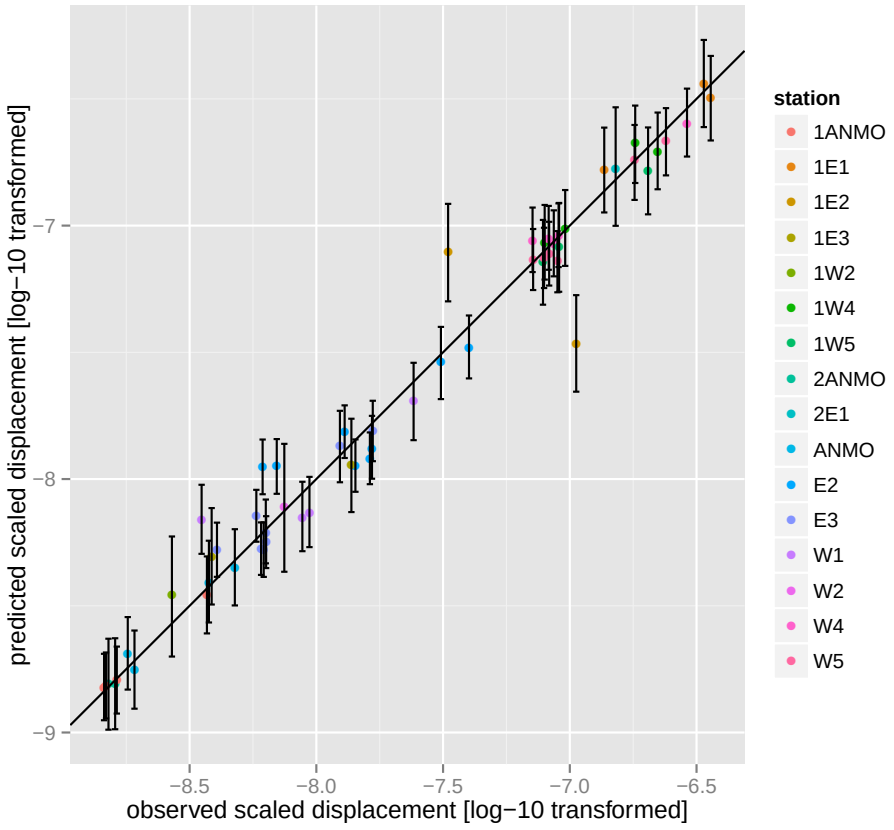

Figure 16: Same as Figure 15, except with the prediction points colored by station ID (rather than event ID).

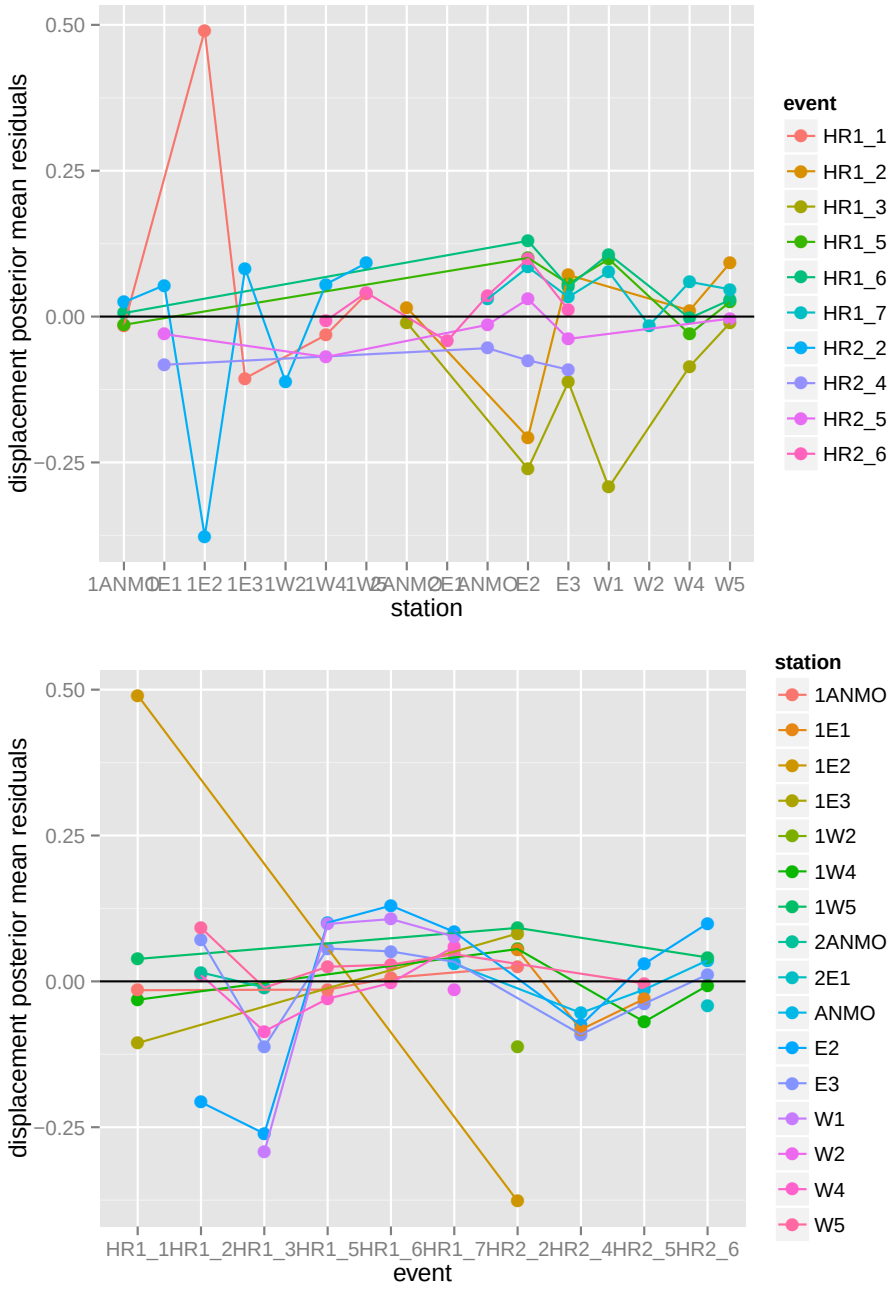

Figure 17: The posterior mean residuals versus (top) seismic station IDs and (bottom) explosion/event IDs from Model 2 from STAN Code 3.

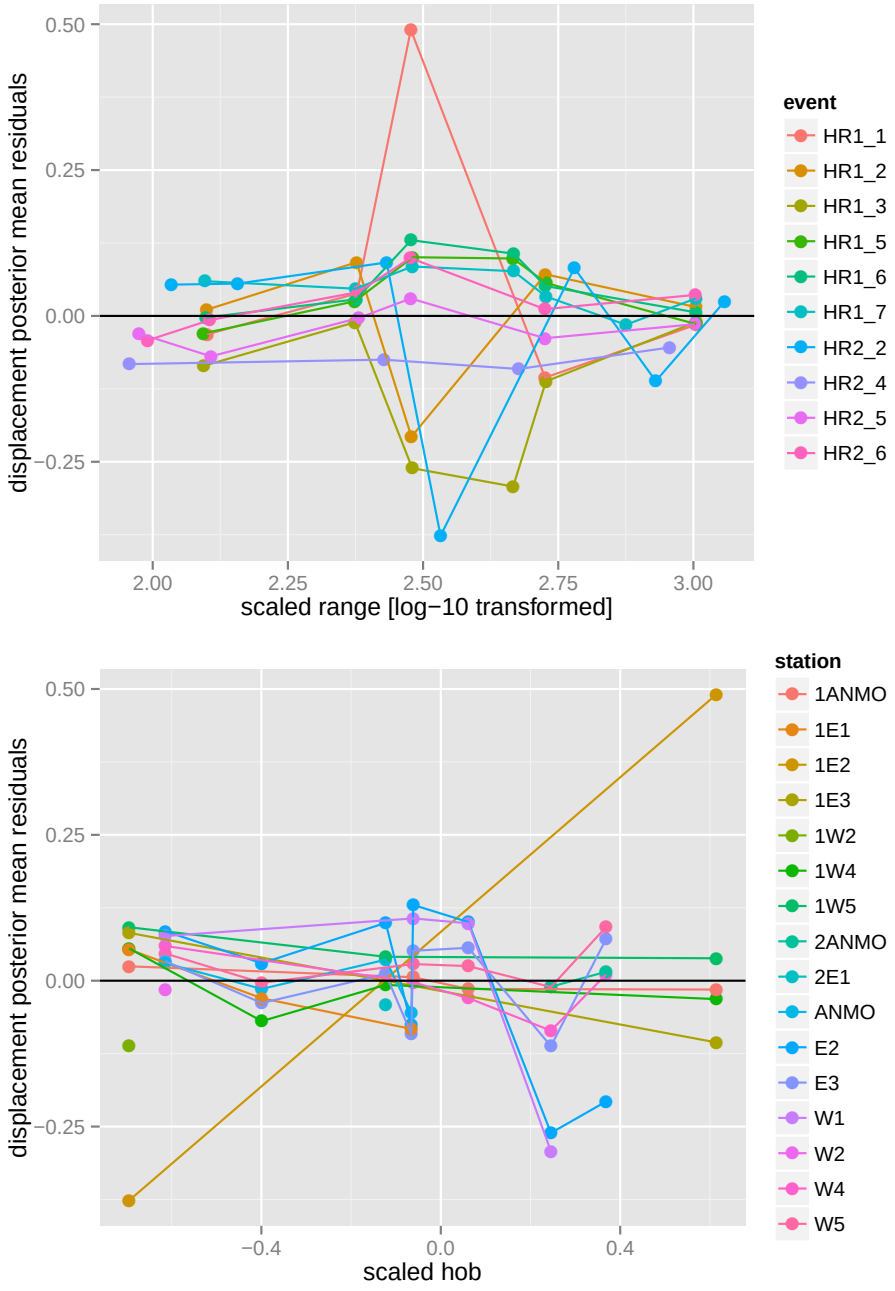

Figure 18: Similar to Figure 17, but with the posterior mean residuals versus (top) the scaled range and (bottom) scaled HOB from Model 2 from STAN Code 3.

4.2 Air Pressure Impulse Training Model

A simple starting model is given in Model 3.

Model 3 A Simple hierarchical Bayesian training model for air pressure impulse.

$$
\begin{aligned}
p(\log_{10} y^s_{aij} \,|\, \hat{y}_{aij}(\beta_a), \sigma_a) &= \text{Gau}(\cdot \mid \log_{10} \hat{y}_{aij}(\beta_a), \sigma_a^2), \\
\log_{10} \hat{y}^2_{aij}(\beta_a) &= \beta_{a1} + \beta_{a2} \log_{10}(r^s_{aij}) \\
&\quad + \beta_{d3} h^s_i + \log_{10}(1 + 10^{10\beta_{a3} h^s_i})/10 \\
p(\beta_{aj}) &\propto 1, \quad \text{for } j = 1, 2, 3 \text{ (non-informative)}, \\
p(\sigma_a^2) &\propto 1/\sigma_a^2, \qquad \text{(non-informative prior)}.
\end{aligned}
$$

4.2.1 Posterior Inference

The STAN code for Model 3 is given in Code 4.

It took roughly 17 seconds to yield the needed STAN C++ executable code and then another 1.4 seconds to run 4 MCMC chains for 5000 iterations (retaining only every 5th iteration of the last 2500 iterations). A posterior summary is given in Table 4 while Figures 19–22 provide graphical summaries. We note the following:

- The MCMC chains are very well mixed (Figure 19).
- There is an obvious explosion/event bias in the residuals, but very little station bias, which is the opposite to the seismic training model.

Code 4 The STAN code for Model 3.

```
## STAN model to train on air pressure data (i.e., known yields and hob)
data {
    ## the seismic data:
    int<lower=0> n;                  ## number of air pressure impulse obs
    vector[n] log10_impulse_scaled;  ## = log10(impulse/yield^(1/3))
    vector[n] log10_range_scaled;    ## = log10(range/yield^(1/3))
    vector[n] hob_scaled;            ## = hob/yield^(1/3)
    vector<lower=0>[n] stdev;   ## = estimated stdev of impulse obs
    int<lower=0> n_s;           ## = # of stations
    int<lower=0> i_s[n];        ## index to station, between 1 and n_s
}

parameters {
    vector[3] beta;            ## the param of the impulse model
    real<lower=0> sigma;       ## the stdev
}

transformed parameters {
    vector[n] mu;
    for(i in 1:n) {
        mu[i] <- beta[1] + beta[2]*log10_range_scaled[i] + beta[3]*hob_scaled[i] +
            log10(1.0 + pow(10.0, 10.0*beta[3]*hob_scaled[i]))/10.0;
    }
}

model {
    log10_impulse_scaled ~ normal(mu, sigma);
    increment_log_prob(-2.0*log(sigma));  # p(sigma^2) propto 1/sigma^2
}
```

```
Inference for Stan model: air_training_model_1.
4 chains, each with iter=5000; warmup=2500; thin=5;
post-warmup draws per chain=500, total post-warmup draws=2000.

         mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
beta[1]  2.57    0.01 0.22  2.13  2.42  2.56  2.71  2.99  1866    1
beta[2] -1.25    0.00 0.12 -1.49 -1.33 -1.25 -1.18 -1.02  1886    1
beta[3]  0.87    0.00 0.09  0.68  0.81  0.87  0.93  1.05  1964    1
sigma    0.30    0.00 0.03  0.25  0.28  0.30  0.32  0.36  2000    1

Samples were drawn using NUTS(diag_e) at Fri Oct 24 14:22:51 2014.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

Table 4: Posterior summary statistics for the air pressure impulse training Model 3 from STAN Code 4.

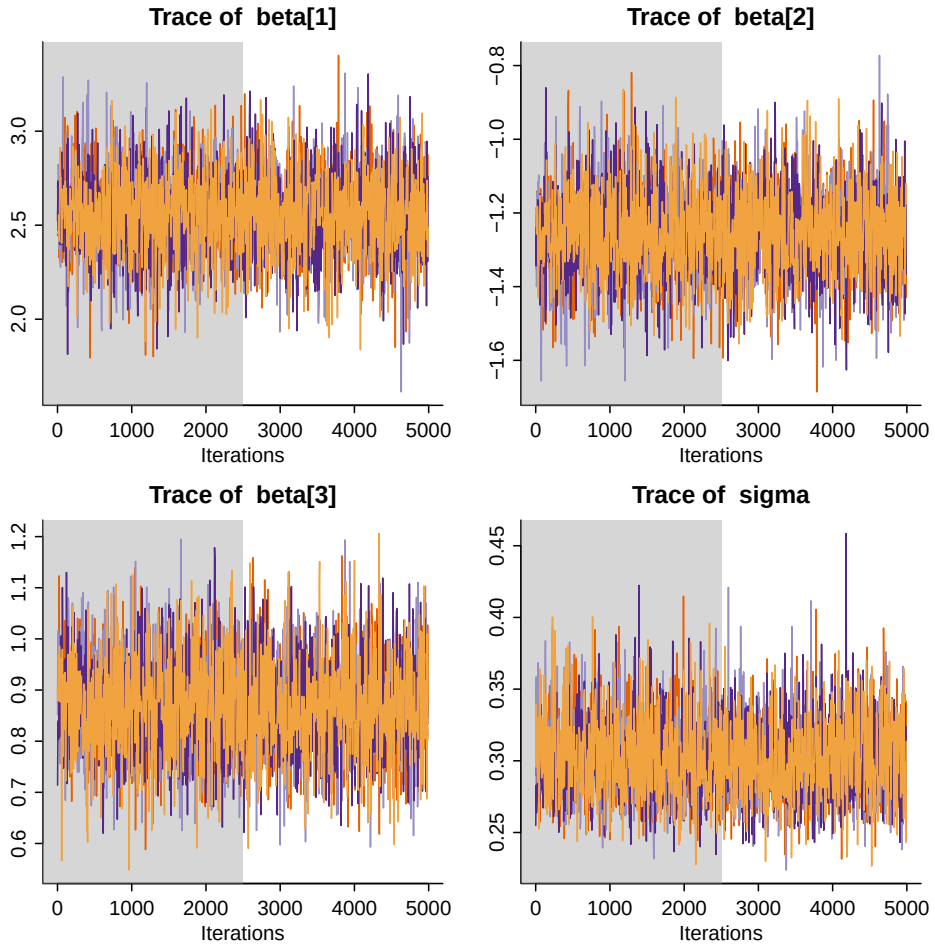

Figure 19: MCMC trace plots for the variables in Model 3 from STAN Code 4.

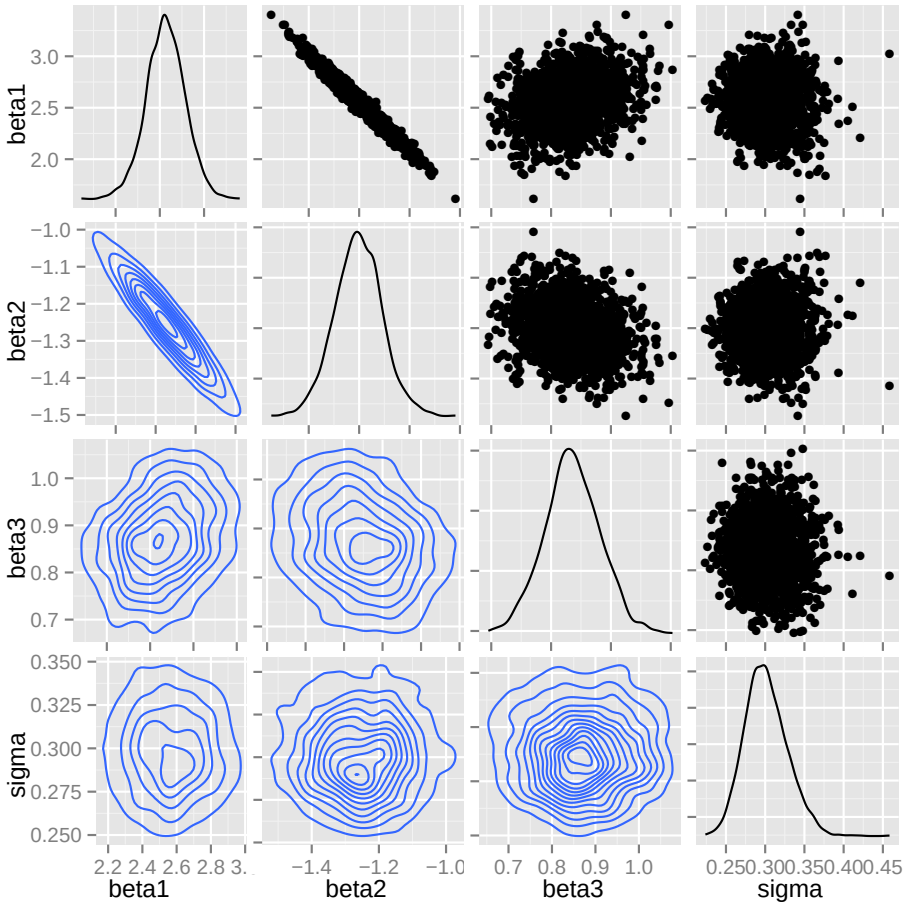

Figure 20: Graphical summary of the posterior distributions of $\beta_{a1}, \ldots, \beta_{a3}$ and σ_a in Model 3 from STAN Code 4.

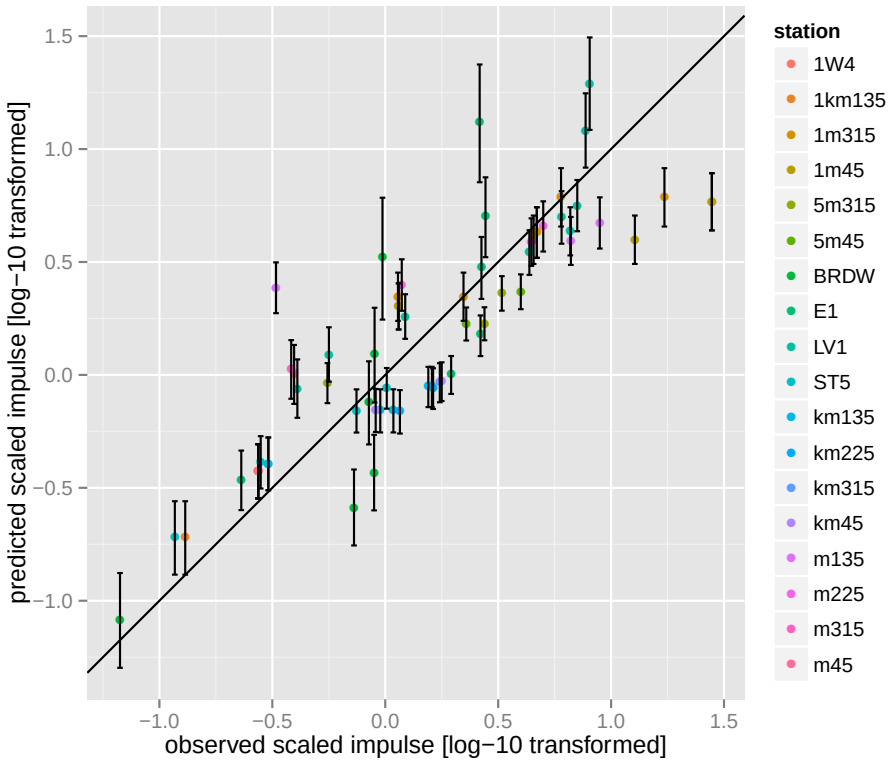

Figure 21: The posterior mean predicted values of $\log_{10} \hat{y}_{aij}(\beta_a)$, along with empirical 95% prediction intervals, for Model 3 from STAN Code 4.

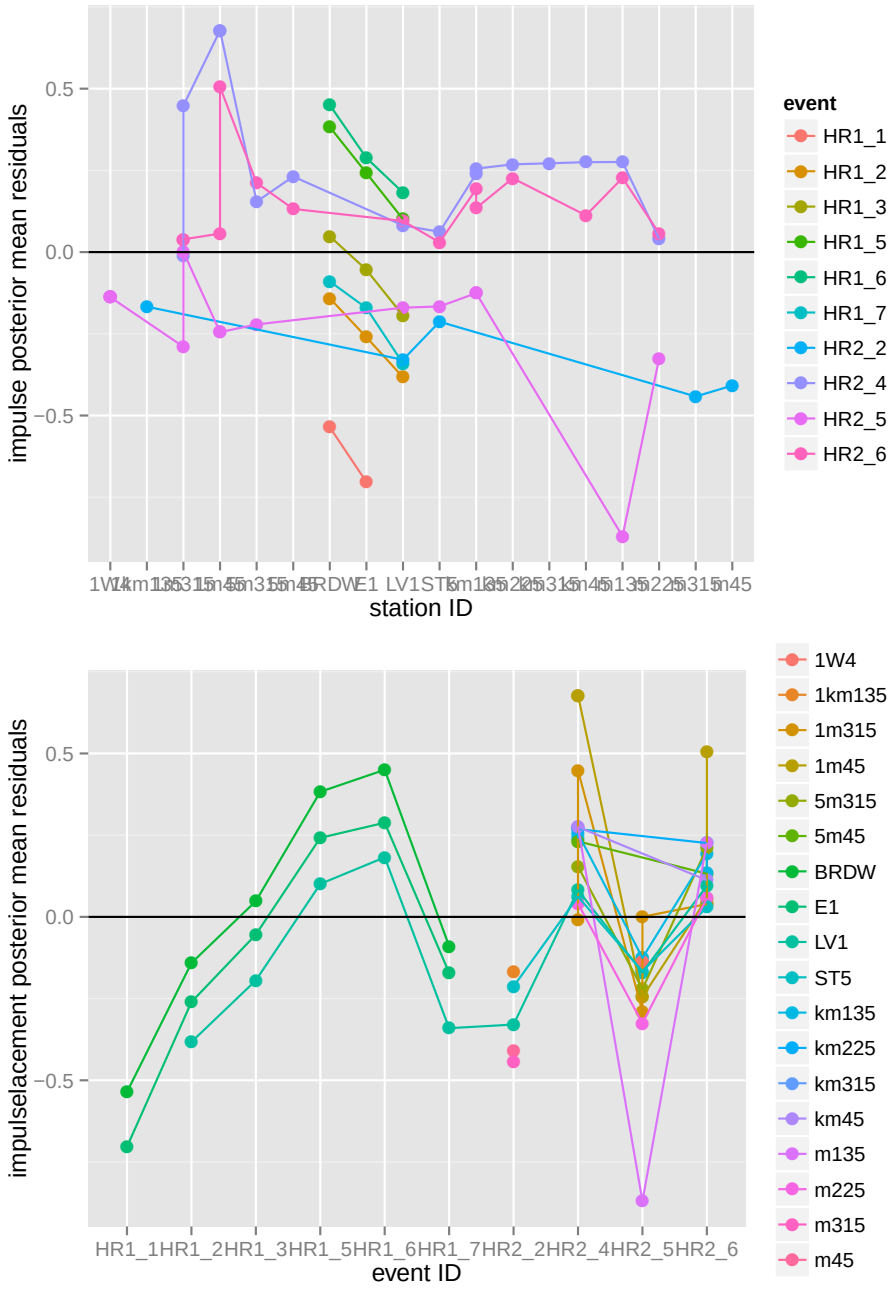

Figure 22: The posterior mean residuals versus (top) meteorology station IDs and (bottom) explosion/event IDs from Model 3 from STAN Code 4.

5 Bayesian Yield and HOB Inversion

We now describe the Bayesian inversion for unknown yield and HOB associated with a new event. We will consider two approaches:

- A yield and HOB Bayesian inversion given trained seismic displacement and air pressure impulse models.
- A joint Bayesian inversion of yield and HOB along with all the parameters that define both the seismic displacement forward model and the air pressure impulse forward model.

In the first approach, the yield and HOB for a new event are treated as unknown and stochastic, with prior PDFs (typically relatively vague) placed on them. The prior PDFs for the forward model parameters (β's, etc.) are taken as the *posterior* from the training phase, so these are typically "tight" (informative) PDFs. In contrast, in the second approach, both the forward model parameters and the yields and HOBs of multiple events are treated as unknown, but the prior PDFs for those events with "known" yield and HOB (i.e., the training data) are taken as very "informative" (e.g., a point mass at the "true" value), while the prior PDF for the event with unknown yield and HOB is taken as "flat".

5.1 Yield and HOB Inversion Given Trained Seismic Displacement and Air Pressure Impulse Models

The basic inversion model, corresponding to the seismic displacement model outlined in Model 1 and the air pressure impulse model outlined in Model 3, is provided in Model 4.

An MCMC algorithm to sample from the posterior distribution of Model 4 consists of three main sampling steps:

1. update the unknown yield and HOB,
2. update the training parameters of the seismic displacement model, and
3. update the training parameters of the air pressure impulse model.

An MH random walk algorithm to sample yield and HOB (w and h) is outlined in Algorithm 11, which is conditional on a particular realization of the parameters of the seismic displacement and the air pressure impulse training models, denoted as $\theta_d(\beta_d, \sigma_d)$ and $\theta_a = (\beta_a, \sigma_a)$ for Models 1 and 3, respectively.

Model 4 A Bayesian inversion model for unknown yield and HOB (w and h), associated with a new event with observed seismic displacement data y_{dj} and air pressure impulse data y_{aj}. The prior PDFs of the parameters for the forward models is taken as the posterior PDFs of the parameters from earlier Bayesian training of both models.

Seismic displacement data model:

$$\begin{aligned} p(\log_{10} y_{dj} \,|\, \hat{y}_{dj}(\beta_d, w, h), \sigma_d) &= \mathrm{Gau}(\cdot \mid \log_{10} \hat{y}_{dj}(\beta_d, w, h), \sigma_d^2), \\ \log_{10} \hat{y}_{dj}(\beta_d, w, h) &= \tfrac{1}{3} \log_{10}(w) + \beta_{d1} + \beta_{d2} \log_{10}(r_{dj}/w^{1/3}) \\ &\quad + \beta_{d3} \tanh(\beta_{d5}(h/w^{1/3}) - \beta_{d4}))/\beta_{d5}, \end{aligned}$$

Air pressure impulse data model:

$$\begin{aligned} p(\log_{10} y_{aj} \,|\, \hat{y}_{aj}(\beta_a, w, h), \sigma_a) &= \mathrm{Gau}(\cdot \mid \log_{10} \hat{y}_{aj}(\beta_a, w, h), \sigma_a^2), \\ \log_{10} \hat{y}_{aj}(\beta_a, w, h) &= \tfrac{1}{3} \log_{10}(w) + \beta_{a1} + \beta_{a2} \log_{10}(r_{aj}/w^{1/3}) \\ &\quad + \beta_{d3} h/w^{1/3} + \log_{10}(1 + 10^{10\beta_{a3} h/w^{1/3}})/10. \end{aligned}$$

Prior for forward models:

$$\begin{aligned} \pi(\theta_d) &= p(\theta_d \,|\, \{y_{dij}\}) = \text{posterior training PDF from seismic Model 1} \\ \pi(\theta_a) &= p(\theta_a \,|\, \{y_{aij}\}) = \text{posterior training PDF from air Model 3.} \end{aligned}$$

where $\theta_d = (\beta_{d1}, \ldots, \beta_{d5}, \sigma_d)$ and $\theta_a = (\beta_{a1}, \beta_{a2}, \beta_{a3}, \sigma_a)$; $\{y_{dij}\}$ = the seismic training data and $\{y_{aij}\}$ = the air pressure training data.

Prior for yield and HOB:

$$\begin{aligned} p(w) &= \mathrm{Gau}(w \,|\, M_w, V_w)_{[0,\infty)}, \\ p(h) &= \mathrm{Gau}(h \,|\, M_h, V_h). \end{aligned}$$

Note that to sample a new realization of the parameters of the training models, we do not need to have a closed-form expression for the posterior PDFs of those parameters from the training phase (denoted as $\pi(\theta_d) = p(\theta_d \,|\, \{y_{dij}\})$ and $\pi(\theta_a) = p(\theta_a \,|\, \{y_{aij}\})$ in Model 4). However, we have pos-

terior realizations from both PDFs, which we denote as

$$\tilde{\theta}_d^{(1)}, \ldots, \tilde{\theta}_d^{(N)} = \text{posterior realizations from } \pi(\theta_d) \quad \text{and}$$
$$\tilde{\theta}_a^{(1)}, \ldots, \tilde{\theta}_a^{(N)} = \text{posterior realizations from } \pi(\theta_a)$$

This means that if we need a realization from either of these two PDFs, we can simply draw at random one of the existing realizations. Algorithm 12 provides an MH algorithm to sample a new realization of the seismic displacement model parameters, $\theta_d = (\beta_d, \sigma_d)$. The proposal distribution for θ_d is simply taken as the posterior PDF from the training phase, $\pi(\theta_d)$. Therefore, when computing the acceptance ratio for the new move, the contribution from the prior (in this case $\pi(\theta_d)$) and the proposal distribution cancel each other out and one is only left with the likelihood ratio of the new displacement observations ($\{y_{dj}\}$).

An MH algorithm to sample new air pressure impulse model parameters, $\theta_a = (\beta_a, \sigma_a)$, is analogous to MH_$\theta_d(\theta_d^{(t)}; w^{(t)}, h^{(t)})$ procedure given in Algorithm 12 and therefore is not specifically outlined, but we shall refer to it as Procedure MH_$\theta_a(\theta_a^{(t)}; w^{(t)}, h^{(t)})$.

A complete MH algorithm to sample $(w, h, \theta_d, theta_a)$ simply iterates between updating (1) w and h, (2) θ_d, and (3) θ_a, as mentioned previously.

6 A Mixture Model for Seismic Displacement

Experiments conducted at vairous soft-rock and hard-rock sites during the period 2007-2014 suggest that scaled displacement as a function of scaled HOB may be clustered according to the type of rock, presence of fractures in the rock and other factors. As shown in Figure 23, there are 2 clusters of displacement values at higher values of HOB: one for limestone and the other for alluvium and granite. If that is the case, the forward model in (2) needs to be modified. To simplify notation, rewrite the PDF of scaled displacement as follows:

$$y_{ij}^s \sim N(f(\boldsymbol{\beta}, r_{dij}^s, h_i^s), \kappa^{-1}), \tag{8}$$

where $f(\boldsymbol{\beta}, r_{dij}^s, h_i^s)$ is the right-hand-side of (2) with $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$.

Now, to account for the fact that there are possibly clusters of forward models for displacement, modify (8) as follows:

$$y_{ij}^s \sim N(f(\boldsymbol{\beta}_{\delta_{ij}}, r_{dij}^s, h_i^s), \kappa_{\delta_{ij}}^{-1}), \tag{9}$$

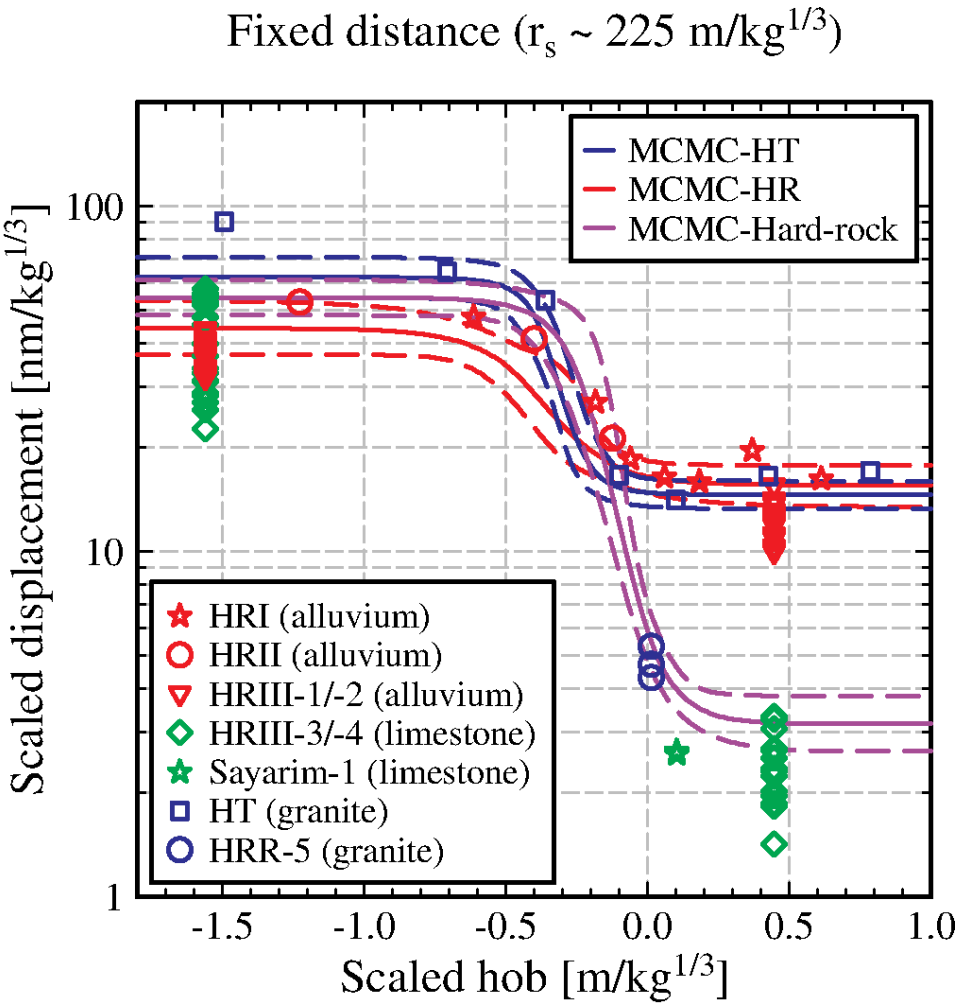

Figure 23: A plot of scaled displacement versus scaled HOB, observed at various soft- and hard-rock sites 2007-2014, with colors and symbols indicating locations and rock types. The dashed lines represent the forward model scaled displacement predictions (means and 95% prediction intervals).

where δ_{ij} is the indicator for the cluster. The prior distribution for δ_{ij} is given by

$$\delta_{ij} \sim \text{Multinomial}(\boldsymbol{\pi}^{(\mathbf{0})}) \tag{10}$$

with $\boldsymbol{\pi}^{(\mathbf{0})} = (\pi_1^{(0)}, ..., \pi_K^{(0)})$ the vector of probabilities of being in each of K clusters, and $\sum_{k=1}^{K} \pi_k^{(0)} = 1$. Note that if there are only 2 clusters, this simplifies to Bernouilli($\pi_1^{(0)}$) where $\pi_1^{(0)}$ is the probability of being in the first cluster.

The same priors as before can be put on $\boldsymbol{\beta}_{\delta_{ij}}$ and $\kappa_{\delta_{ij}}$ for all δ_{ij}, but one can also introduce constraints. For example, in the case of two clusters, one might know that $\beta_{5_{\delta_{ij}=1}} < \beta_{5_{\delta_{ij}=2}}$.

The full likelihood for the mixture model specified above is as follows:

$$\prod_{i,j} \left[\sum_{k=1}^{K} \pi_k l_k(y_{ij}^s; \boldsymbol{\beta}, \kappa) \right], \tag{11}$$

where l_k is the likelihood associated with just the kth cluster (i.e., the Gaussian likelihood corresponding to Eq. (8) above for a given cluster). This full likelihood can be hard to work with directly, especially if the number of clusters K is not very small.

An alternative is to modify the MCMC sampler as follows. At each iteration t of the MCMC,

1. Sample $\delta_{ij}^{(t)}$ from its full conditional multinomial distribution given by

$$\pi_k^{(t)} = p(\delta_{ij}^{(t)} = k) \propto \pi_k^{(t-1)} l_k(y_{ij}^s; \boldsymbol{\beta}^{(t-1)}, \kappa^{(t-1)}) \tag{12}$$

   with $\pi_k^{(0)}$ as given above and $l_k(y_{ij}^s; \boldsymbol{\beta}^{(t-1)}, \kappa^{(t-1)})$ the Gaussian likelihood in the kth cluster with the most recent estimates of $\boldsymbol{\beta}$ and κ.

2. Then sample $\boldsymbol{\beta}$ and κ as before, but separately for each cluster. In particular, compute the log-acceptance ratios for $\boldsymbol{\beta}$ and κ as usual (see, step 5 of Algorithm 2 and step 3 of Algorithm 3, for $\boldsymbol{\beta}$ and κ, respectively), but separately for groups of observations with $\delta_{ij}^{(t)} = k$ for each value of $k = 1, ..., K$.

7 Future Work

Two aspects of the framework presented here need to be addressed in future work. One is that currently, the PDF for the observations conditional on

predictions in (4) does not make a distinction between the forward model error and observation error. There is just one error term (reflected in the parameter κ) that captures both of these sources of uncertainty. However, if one has information about the relative contributions of the two sources, one could build that into the framework by explicitly modeling each one. The prior distributions for each error term would need to be somewhat informative to ensure identifiability (i.e., that the two terms can be distinguished from one another).

The other need that must be addressed in the present framework is the implementation of appropriate tests of convergence of the MCMC algorithms implemented here. In particular, while there are very well established convergence tests for univariate target distributions, when the distributions of interest are multivariate, as is the case here, such tests are less straightforward. However, from literature review (see, e.g., Cowles and Carlin (1996) for an overview), we have identified a potential candidate for such a test that is fairly simple to implement and does not require satisfying very stringent conditions. It is covered in Brooks and Gelman (1998) and provides a multivariate extension of the well-known Gelman and Rubin method in the univariate setting Gelman and Rubin (1992). In the univariate version, one calculates a statistic that is a linear function of the ratio of the between-chain to the within-chain variance. If this statistic is large, it indicates that the chains have not yet converged. In the multivariate setting the between-chain and within-chain variances are replaced with the corresponding covariance matrices, and in place of the ratio of the variances one uses the largest eigenvalue of the matrix obtained by multiplying the inverse of the within-chain covariance matrix by the between-chain covariance matrix.

Algorithm 11 An MH random walk algorithm to sample the yield and HOB (w and h) for a new event, given a particular realization of the seismic displacement model, $\theta_d = (\beta_d, \sigma_d)$, and air pressure impulse model, $\theta_a = (\beta_a, \sigma_a)$.

procedure MH_RW_$wh(w^{(t)}, h^{(t)}$; θ_d, θ_a, τ_w, $\tau_h)$

Propose new yield: $w^* \leftarrow |w^{(t)} + \epsilon|$, where $\epsilon \sim \text{Gau}(0, \tau_w^2)$

Propose new HOB: $h^* \leftarrow h^{(t)} + \epsilon$, where $\epsilon \sim \text{Gau}(0, \tau_h^2)$

Compute the seismic displacement log-likelihood ratio:

$$\begin{aligned}\log \rho_d &\leftarrow \sum_{j=1}^{n_d} \Big(\log \text{Gau}(\log_{10} y_{dj} \mid \hat{y}_{dj}^*, \sigma_d^2) - \log \text{Gau}(\log_{10} y_{dj} \mid \hat{y}_{dj}^{(t)}, \sigma_d^2) \Big) \\ &= -\frac{1}{2\sigma_d^2} \sum_{j=1}^{n_d} \Big((\log_{10} y_{dj} - \log_{10} \hat{y}_{dj}^*)^2 - (\log_{10} y_{dj} - \log_{10} \hat{y}_{dj}^{(t)})^2 \Big),\end{aligned}$$

where $\hat{y}_{dj}^* \leftarrow \hat{y}_{dj}(\beta_d, w^*, h^*)$ and $\hat{y}_{dj}^{(t)} \leftarrow \hat{y}_{dj}(\beta_d, w^{(t)}, h^{(t)})$

Compute the air pressure impulse log-likelihood ratio:

$$\begin{aligned}\log \rho_a &\leftarrow \sum_{j=1}^{n_a} \Big(\log \text{Gau}(\log_{10} y_{aj} \mid \hat{y}_{aj}^*, \sigma_a^2) - \log \text{Gau}(\log_{10} y_{aj} \mid \hat{y}_{aj}^{(t)}, \sigma_a^2) \Big) \\ &= -\frac{1}{2\sigma_a^2} \sum_{j=1}^{n_a} \Big((\log_{10} y_{aj} - \log_{10} \hat{y}_{aj}^*)^2 - (\log_{10} y_{aj} - \log_{10} \hat{y}_{aj}^{(t)})^2 \Big),\end{aligned}$$

where $\hat{y}_{aj}^* \leftarrow \hat{y}_{aj}(\beta_a, w^*, h^*)$ and $\hat{y}_{aj}^{(t)} \leftarrow \hat{y}_{aj}(\beta_a, w^{(t)}, h^{(t)})$

Compute the log-prior ratio:

$$\begin{aligned}\log \rho_0 &\leftarrow \Big(\log \text{Gau}(w^* \mid M_w, V_w) - \log \text{Gau}(w^{(t)} \mid M_w, V_w) \Big) \\ &\quad + \Big(\log \text{Gau}(h^* \mid M_h, V_h) - \log(h^{(t)} \mid M_h, V_h) \Big) \\ &= -\frac{1}{2V_w} \Big((w^* - M_w)^2 - (w^{(t)} - M_w)^2 \Big) \\ &\quad - \frac{1}{2V_a} \Big((h^* - M_h)^2 - (h^{(t)} - M_h)^2 \Big)\end{aligned}$$

Compute the log-acceptance rate: $\log \alpha = \log \rho_d + \log \rho_a + \log \rho_0$

Generate $u \sim \text{Unif}(0, 1)$ and **if** $\alpha \geq u$ **then** $(w^{(t+1)}, h^{(t+1)}) \leftarrow (w^*, h^*)$ **else** $(w^{(t+1)}, h^{(t+1)}) \leftarrow (w^{(t)}, h^{(t)})$

return $(w^{(t+1)}, h^{(t+1)})$

end procedure

Algorithm 12 An MH algorithm to sample a new realization of $\theta_d = (\beta_d, \sigma_d)$, given a particular realization of the yield and HOB (w and h).

procedure MH_$\theta_d(\theta_d^{(t)};\, w^{(t)},\, h^{(t)})$

Propose a new θ_d: $\theta_d^* \leftarrow \tilde{\theta}_d^{(b^*)}$, that is, $(\beta_d^*, \sigma_d^*) \leftarrow (\tilde{\beta}_d^{(b^*)}, \tilde{\sigma}_d^{(b^*)})$, where $b^* \sim \text{Unif}\{1, 2, \ldots, N\}$

Compute the log-likelihood ratio:

$$\begin{aligned}
\log \rho_d &\leftarrow \sum_{j=1}^{n_d} \left(\log \text{Gau}(\log_{10} y_{dj} \,|\, \hat{y}_{dj}^*, (\sigma_d^*)^2) - \log \text{Gau}(\log_{10} y_{dj} \,|\, \hat{y}_{dj}^{(t)}, (\sigma_d^{(t)})^2) \right) \\
&= \left(-\frac{1}{2} n_d \log((\sigma_d^*)^2) - \frac{1}{2(\sigma_d^*)^2} \sum_{j=1}^{n_d} (\log_{10} y_{dj} - \log_{10} \hat{y}_{dj}^*)^2 \right) \\
&\quad - \left(-\frac{1}{2} n_d \log((\sigma_d^{(t)})^2) - \frac{1}{2(\sigma_d^{(t)})^2} \sum_{j=1}^{n_d} (\log_{10} y_{dj} - \log_{10} \hat{y}_{dj}^{(t)})^2 \right),
\end{aligned}$$

where $\hat{y}_{dj}^* \leftarrow \hat{y}_{dj}(\beta_d^*, w^{(t)}, h^{(t)})$ and $\hat{y}_{dj}^{(t)} \leftarrow \hat{y}_{dj}(\beta_d^{(t)}, w^{(t)}, h^{(t)})$

Compute the log-acceptance rate: $\log \alpha = \log \rho_d$

Generate $u \sim \text{Unif}(0, 1)$ and **if** $\alpha \geq u$ **then** $\theta_d^{(t+1)} \leftarrow \theta_d^*$ **else** $\theta_d^{(t+1)} \leftarrow \theta_d^{(t)}$

return $\theta_d^{(t+1)}$

end procedure

A Gibbs Sampler

One way to view a Gibbs sampler is as a Metropolis-Hastings sampler which produces a proposal that is always accepted (i.e., $\alpha = 1$ in Algorithm 1). The is accomplished by taking the proposal distribution as the full conditional distribution (if one can sample from it). For example, if the target distribution is (6), and if the full conditional distribution of κ, given by

$$\pi(\kappa \,|\, \beta) = p(\kappa \,|\, \beta, y) \propto p(y \,|\, \hat{y}(\beta), \kappa) p(\kappa),$$

can be sampled from, then it can be used as the proposal distribution κ in Algorithm 5 and it can be shown that it yields a proposal that is always accepted. Note that the proposal does not depend on the previous value of κ in the Markov chain.

Two useful full conditional distributions we will come across are the normal-normal conjugate relationship and the normal-gamma relationship. In both cases, assume we the following setup:

$$\begin{aligned} r_i &\sim \text{Gau}(x_i\beta, 1/\kappa), \text{ for } i = 1, \ldots, n, \\ \beta &\sim \text{Gau}(\mu_0, 1/\kappa_0), \\ \kappa &\sim \text{Gam}(a_0, b_0). \end{aligned}$$

That is, our "data" (the r_is) are assumed to be normally distributed with the mean given by $x_i\beta$, where the x_i's are known, and variance $1/\kappa$. We then have a normal prior for β and gamma prior for κ.

Given the setup above, the full conditional distribution of κ, $\pi(\kappa \,|\, \beta)$ is a gamma distribution and the full conditional distribution of β is a normal distribution. The details can be found in Gelman et al. (2013).

References

C. Andrieu and J. Thoms. A tutorial on adaptive MCMC. *Statistical Computing*, 18:343–373, 2008.

S. Brooks and A. Gelman. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7:434–455, 1998.

M. C. Cowles and B. Carlin. Markov Chain Monte Carlo convergence diagnostics: A comparative review. *Journal of the American Statistical Association*, 91:883–904, 1996.

S. R. Ford, A. J. Rodgers, H. X., D. C. Templeton, P. Harben, W. Foxall, and R. E. Reinke. Partitioning of seismoacoustic energy and estimation of yield and height-of-burst/depth-of-burial for near-surface explosions. *Bulletin of the Seismological Society of America*, 104(2):608–623, 2014. doi: 10.1785/0120130130.

W. Foxall, R. Marrs, E. Lenox, R. Reinke, D. Seastrand, J. Bonner, K. Mayeda, and C. Snelson. The HUMBLE REDWOOD seismic/acoustic coupling experiments: Joint inversion for yield using seismic, acoustic and crater data. *Seismological Research Letters*, 81:315, 2010.

A. Gelman and D. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7:457–511, 1992.

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. Hapman and Hall/CRC, 3nd edition, 2013.

C. P. Robert and G. Casella. *Monte Carlo Statistical Methods (Second Edition)*. Springer, 2004.